Antti Pakarinen

Multi-core platforms for audio and multimedia coding algorithms in telecommunications

School of Electrical Engineering

Thesis submitted for examination for the degree of Master of Science in Technology.

Kirkkonummi 14.9.2012

Thesis supervisor: Prof. Vesa Välimäki

Thesis instructor: M.Sc. (Tech.) Jussi Pekonen

Aalto University
School of Electrical
Engineering

AALTO UNIVERSITY
SCHOOL OF ELECTRICAL ENGINEERING

ABSTRACT OF THE
MASTER'S THESIS

Author: Antti Pakarinen

Title: Multi-core platforms for audio and multimedia coding algorithms in telecommunications

Date: 14.9.2012 Language: English Number of pages: 9+63

Department of Signal processing and Acoustics

Professorship: Acoustics and Audio Signal processing Code: S-89

Supervisor: Prof. Vesa Välimäki

Instructor: M.Sc. (Tech.) Jussi Pekonen

Multimedia coding algorithms used in telecommunications evolve constantly. Benefits and properties of two new hybrid audio codecs (USAC, Opus) were reviewed on a high level as a literature study. It was found that both have succeeded well in subjective sound quality measurements. Tilera TILEPro64-multicore platform and a related software library was evaluated in terms of performance in multimedia coding. The performance in video coding was found to increase with the number of processing cores up to a certain point. With the tested audio codecs, increasing the number of cores did not increase coding performance. Additionally, multicore products of Tilera, Texas Instruments and Freescale were compared. Development tools of all three vendors were found to have similiar features, despite the differences in hardware architectures.

Keywords: Multimedia, audio, video, coding algorithms, multicore

AALTO-YLIOPISTO
SÄHKÖTEKNIIKAN KORKEAKOULU

DIPLOMITYÖN
TIIVISTELMÄ

Tekijä: Antti Pakarinen

Työn nimi: Moniydinsuorittimet audion ja multimedian koodausalgoritmeille tietoliikenteessä

Päivämäärä: 14.9.2012 Kieli: Englanti Sivumäärä: 9+63

Signaalinkäsittelyn ja akustiikan laitos

Professuuri: Akustiikka ja äänenkäsittely Koodi: S-89

Valvoja: Prof. Vesa Välimäki

Ohjaaja: DI Jussi Pekonen

Tietoliikenteessä käytettävät multimedian koodausalgoritmit eli koodekit kehittyvät jatkuvasti. USAC ja Opus ovat uusia, sekä puheelle että musiikille soveltuvia audiokoodekkeja. Molemmat ovat sijoittuneet korkealle koodekkien äänenlaatua vertailevissa tutkimuksissa. Näiden keskeisiä ominaisuuksia käsitellään kirjallisuuskatsaukseen perustuen. Varsinkin HD-tasoisen videon käsittelyssä käytettävät koodekit vaativat suurta laskentatehoa. Tilera TILEPro64 -moniydinsuorittimen ja sille optimoitujen multimediakoodekkien suorituskykyä testattiin tarkoitukseen suunnitelluilla tietokoneohjelmilla. Tulokset osoittivat, että suoritinytimiä lisättäessä videon koodausalgoritmien suoritusnopeus kasvaa tiettyyn rajaan asti. Testatuilla äänen koodausalgoritmeillä ytimien lisääminen ei parantanut suoritusnopeutta. Tileran moniydinratkaisuja verrattiin lopuksi Freescalen ja Texas Instrumentsin moniydinratkaisuihin. Huolimatta eroista laitteistoarkkitehtuureissa, kyseisten toimittajien kehitystyökaluissa todettiin olevan paljon samoja piirteitä.

Avainsanat: Multimedia, audio, video, koodausalgoritmit, moniydin

Preface

First of all, thank you Oy L M Ericsson Ab and Kalle Lindroos for arranging the topic and funding for this thesis. I want to thank Jussi Pekonen and Dietmar Fiedler for guidance and Vesa Välimäki for supervising. Also, special thanks to my teammates at Ericsson Finland during the summer of 2012, pupu, Chris, Micke, Heikki and Kari.

Finally, my wife Tiina and folks back home, thank you for all the love and support. Again.

Helsinki, 14.9.2012

Antti Pakarinen

Contents

Abbreviations

2G	Second Generation network
3G	Third Generation network
3GPP	Third Generation Partnership Project
4G	Fourth Generation network
AAC	Advanced Audio Coding
ACELP	Algebraic Code-excited Linear Prediction
AMR	Adaptive Multirate
AMR-NB	Adaptive Multirate Narrowband
AMR-WB	Adaptive Multirate Wideband
BGF	Border Gateway Function
CABAC	Context-based Adaptive Binary Arithmetic Coding
CELP	Code Excited Linear Prediction
CELT	Constrained-Energy Lapped Transform
CPU	Central Processing Unit
DSP	Digital Signal Processing
DFT	Discrete Fourier Transform
DTMF	Dual-Tone Multi-Frequency
FAC	Forward Aliasing Cancellation
FD	Frequency Domain coding
FIR	Finite Impulse Response
FPS	Frames Per Second
GPP	General Purpose Processor
HD	High Definition
HDVICP	High Definition Video Coprocessor
HE-AAC	High Efficiency Advanced Audio Coding
IDE	Integrated Development Environment
IETF	Internet Engineering Task Force
IM-MGW	IP Multimedia Gateway
ITU	International Telecommunications Union
ITU-T	Telecommunication Standardization Sector
IMS	IP Multimedia Subsystem

IP	Internet Protocol
IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6
ISO	International Organization for Standardization
LPC	Linear Predictive Coding
LTE	Long Term Evolution
M2M	Machine-to-Machine
MDCT	Modified Discrete Cosine Transform
MDE	Multicore Development Environment
MGC	Media Gateway Controller
MIPS	Million Instructions Per Second
MMACS	Million Multiply-And-Accumulates per Second
MOS	Mean Opinion Score
MP3	MPEG-1 Audio Layer 3
MPEG	Moving Picture Experts Group
MRFC	Multimedia Resource Function Controller
MRFP	Multimedia Resource Function Processor
MRS	Media Resource System
MSC-S	Mobile Switching Center Server
MUSHRA	Multiple Stimuli with Hidden Reference and Anchor
NNI	Network-to-Network Interface
PCM	Pulse Code Modulation
PESQ	Perceptual Speech Quality
PLMN	Public Land Mobile Network
PSTN	Public Switched Telephone Network
RAM	Random Access Memory
SDK	Software Development Kit
SGC	Session Gateway Controller
SoC	System on Chip
SSL	Secure Sockets Layer
TCX	Transform Codec Excitation
TDAC	Time Domain Alias Cancellation
TDM	Time-Division Multiplexing

TI	Texas Instruments
TLS	Transport Layer Security
TTR	Tilera Task Runtime
QoS	Quality of Service
UNI	User Network Interface
USAC	Unified Speech and Audio Coding
VoIP	Voice over IP

1 Introduction

Telecommunications is an industry that has been developing rapidly since the mobile telephone was introduced. Technologies used in telecommunications networks evolve constantly and staying on top of this evolution is crucial for companies such as Ericsson. The purpose of this thesis is to serve as an input for Ericsson Finland's internal study on the evolution of multimedia coding algorithms (commonly referred to as codecs) and hardware components that are needed to process such algorithms. The study is related to research and development efforts of Ericsson's Media Resource System (MRS) solution for IP Multimedia Subsystem (IMS) networks. The main research questions addressed by Ericsson Finland were:

- What are the benefits and properties of new hybrid audio codecs?
- What is the performance of a given multicore platform in the context multimedia coding in MRS?
- What benefits does the tested multicore platform offer from the developer's point of view?

Finding answers to these questions forms the structure of this thesis, which will be as follows. After this introduction, the second chapter will give some background on topics that are relevant in the scope of this thesis. This will include a brief introduction to telecommunications industry, followed by the basic concepts of IMS, multimedia coding, and the hardware devices that can handle such coding.

The first research question will be addressed in the third chapter. Two new hybrid audio codecs are reviewed as a literature study. These codecs have recently been standardized and they are examples of the new hybrid method of audio coding that combines the traditionally separate fields of speech and music coding.

The fourth chapter will be a case study, involving the given third party multicore processor platform which was tested in terms of performance in multimedia coding. A set of test applications were developed in order to answer the second research question. The last research question will also be addressed in Chapter four. The usability and quality of the provided development tools and a provided software library for media coding will be assessed.

Chapter five will support the evaluation of the tested multicore platform. It will include a summary of the vendor's other multicore products. In comparison, the properties of two other vendors' multicore products are summarized as well.

2 Background

The research questions in this thesis are related to Ericsson Finland's research and development efforts on their MRS solution for IMS networks. First, the development from the early days of telecommunications to modern IP-based networks such as IMS is briefly reviewed. After the basic concepts of IMS have been discussed, some basic theory behind multimedia coding algorithms is reviewed. The reason why such algorithms are needed in telecommunications is also explained. As the research questions of this thesis are related to multicore platforms, which refers to a single device containing multiple microprocessors, the state of the current microprocessor technology is also briefly reviewed at the end of this section.

2.1 Telecommunications

When information is transferred over distances, the process is called telecommunications. The use of electronic communication devices makes it possible to pass information over significant distances quickly. It started in the mid 1800's, when the first telegraph line was built between Washington and Baltimore by Morse and Vail. In the telegraph, typically short messages were coded as sequences of electrical impulses and sent over a fixed line to the receiver. These impulse sequences were interpreted as letters at the recieving end. Invention of the telephone in 1876 eventually led to a form of telecommunications that utilized speech transmission. [1]

For decades, speech transmission was the only widely used application of telecommunications. Data transmission had been used in research and in the economic life but it was not until the beginning of 1990's when the Internet introduced data traffic over networks to common people [2]. Since then, the development has been rapid. Speech transmission has moved from landlines to mobile devices. In recent years, more applications for mobile devices have emerged than just speech transmission. Modern mobile devices are capable of handling tasks like browsing the Internet, and capturing and streaming media. These devices, commonly referred to as smartphones, are often also integrated with common social media services in various ways.

The kind of services described above have increased the amount of data traffic in mobile networks. 3G networks typically have a good coverage in city areas and service providers commonly offer broadband data as part of a mobile subscription

nowadays. For example in Finland, the amount of data traffic in mobile networks has been growing rapidly in recent years, as shown in Figure 1.

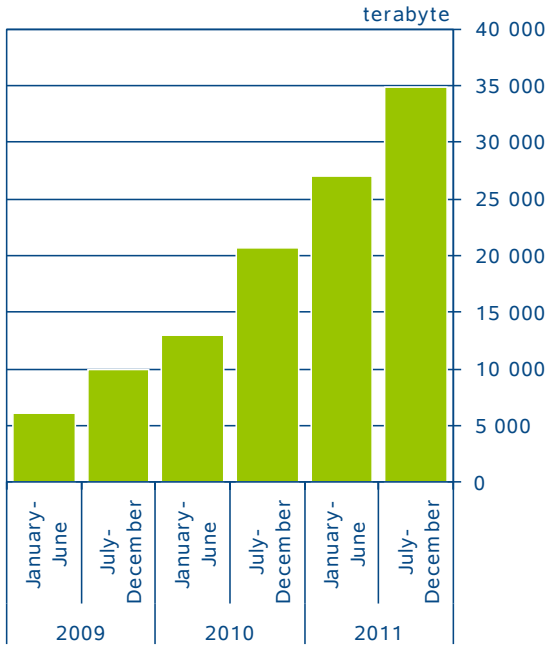

Figure 1: Amount of mobile data traffic in Finland, 2009-2011. Adopted from [3].

At the time of writing this thesis, service providers are deploying the next generation of mobile networks (4G) which will further increase data bandwidths for mobile subscribers. Predictions have been made that in the future, machine to machine (M2M) communications will further increase the need for network capacity worldwide. ”50 billion devices” is a vision originally presented by Ericsson, in which all kinds of devices such as light bulbs, cars, TV’s, and digital cameras will be connected to the Internet by 2020. Telecommunications is therefore predicted to continue its growth as an industry. Different means of communicating over networks are converging and modern networks have to provide the infrastructure for this kind of development. [4]

2.1.1 3GPP

The field of telecommunications is wide and there are numerous organizations and companies who develop devices and services for the industry. Unified practices are needed in order to make them work together as wide networks. There are several organizations worldwide that develop and maintain these unified practices. The 3rd

Generation Partnership Project (3GPP) is a collaboration of the following regional telecommunications standards bodies:

- The European Telecommunications Standards Institute (ETSI)
- The Association of Radio Industries and Businesses, Japan (ARIB)
- The Alliance for Telecommunications Industry Solutions, USA (ATIS)
- China Communications Standards Association (CCSA)
- Telecommunications Technology Association, Korea (TTA)
- Telecommunication Technology Committee, Japan (TTC)

In the field of mobile communications standards, 3GPP has a major role. Despite the name, 3GPP is currently working also on the fourth generation of mobile standards. One of the modern network architectures described by 3GPP is the IMS. [5]

2.1.2 IMS

At present, mobile networks provide users access to the services provided by the Internet via data connections. Some examples of these services are World Wide Web, email, instant messaging, VoIP calls and video conferencing. All of these services, some of which belong to the everyday lifes of millions of people, can be accessed via a mobile data connection. As discussed in section 2.1, the amount of data traffic in mobile networks shows an increasing trend.

From the network operators' point of view, all the different services used through data connections appear as plain data traffic, just bytes been transferred regardless of the service that is been used. IMS provides for operators a framework in which the service usage and quality of service (QoS) can be controlled and monitored. [6]

When it is known to the operator that the user is making a VoIP call or attending a video conference, it can provide a required bandwidth and charge the user based on the type of service and not only in the bytes transferred. When operators can provide QoS in real-time multimedia services, the popularity of such services is more likely to increase. Although VoIP calls and videoconferencing services can be provided over regular data connections to the Internet, QoS might not be guaranteened as reliably when compared to what is possible with IMS. [6]

As in any modern network, making a call over IMS requires involvement of several

nodes within the network. In this context, a node refers to a device or a set of devices that are used to connect different parts of the network together. In the following, some common node types for interworking with IMS networks are briefly reviewed. The role of these nodes in a network is illustrated in Figure 2.

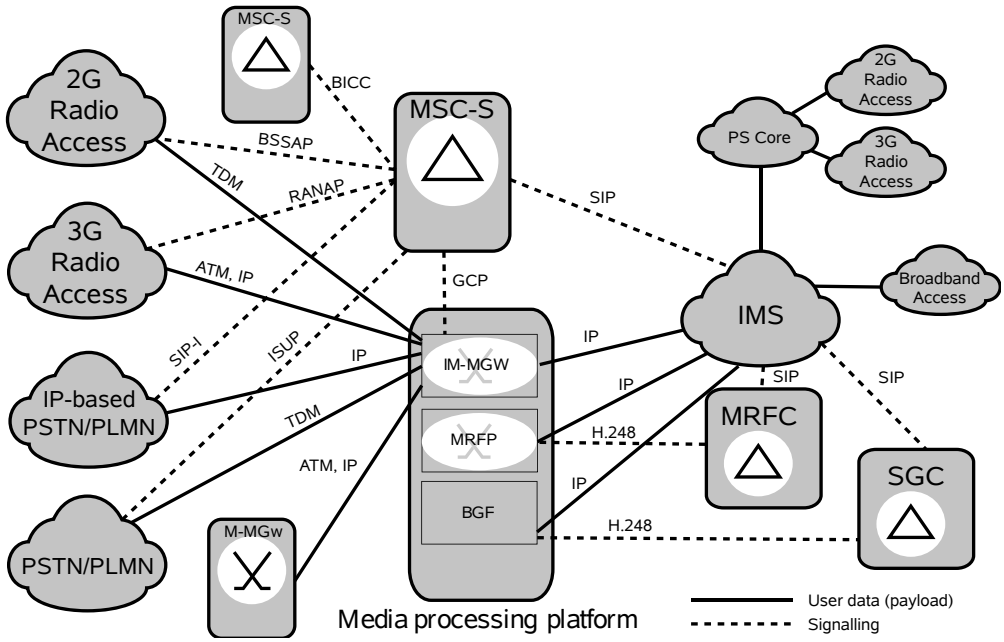

Figure 2: Overview of common multimedia-related node types and their relation to other network nodes [6].

IP Multimedia Gateway (IM-MGW)

Media gateway is a node that handles the transmission of media streams. If necessary, it performs conversions between different media formats and signaling protocols required by different types of networks [7]. IM-MGW handles media stream conversion between IMS and non-IMS networks. For example, IM-MGW is needed when a call is originated within an IMS network and it terminates in another network that uses Time-Division Multiplexing (TDM) instead of IP to transmit speech. In addition to the media conversion, IM-MGW handles the signaling needed to establish a connection between terminals belonging to different types of networks. [6]

Border Gateway Controller (BGF)

Border Gateway Controller (BGF) provides security control at network border towards untrusted networks. It provides media protection against denial of service

attacks and offers firewall functionality. BGF also provides mechanisms for QoS monitoring and controlling. Media conversion between Internet Protocol version 4 (IPv4) and Internet Protocol version 6 (IPv6) can also be done in the BGF. [6]

Multimedia Resource Function Processor (MRFP)

Multimedia Resource Function Processor (MRFP) handles mixing and processing of media streams. For example, MRFP can be used to implement the following functions [6]:

- Building of audio and video conferences
- Detection of dual-tone multi-frequency (DTMF) tones
- Playing of audio announcements and conference tones.

2.2 Basic principles of multimedia coding algorithms

The word codec comes from "coder-decoder" or "compressor-decompressor". In order to transmit media data over a network that has a limited bandwidth, the data has to be compressed. The process of compressing and packing the media information into a suitable format that typically requires less bandwidth is called coding. In the receiving end, the compressed data has to be decompressed by decoding it. Decompression is required in order to play the media back to the user. A certain standard or recommendation of implementing such a compression-decompression process is referred to as a codec.

The encoded multimedia data has to be compressed in a suitable way to be able to transmit it over a given transmission channel, such as an IP network. A flow diagram of a general multimedia transmission event is shown in Figure 3. In this section, basic theory behind multimedia coding is reviewed. The focus will be in audio coding methods in order to support the upcoming discussion on new hybrid audio codecs in Chapter 3.

2.2.1 Audio coding

Codecs used for sound signals have traditionally been divided to speech and audio codecs. "Speech codec" has been used to refer to codecs that are specially designed

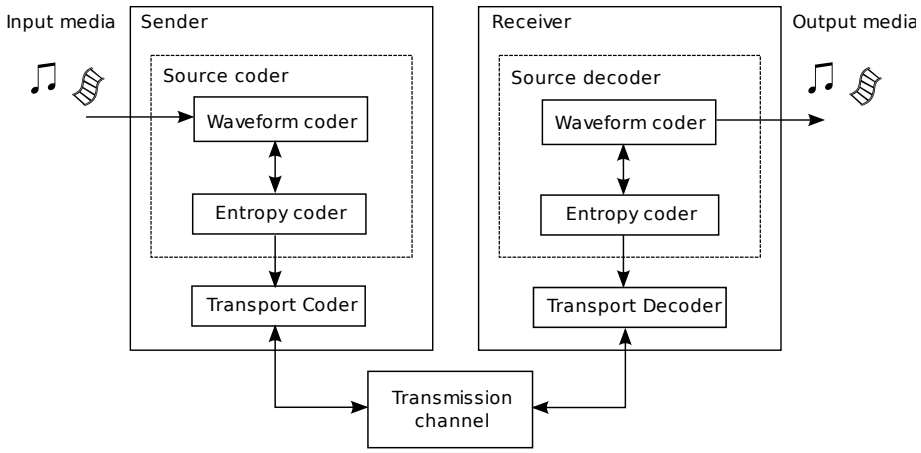

Figure 3: A simplified flow diagram of multimedia transmission. Adapted from [9].

for voice signals. "Audio codec" has been used to refer to all other sound related codecs. In this thesis, "audio codec" refers to a coding scheme that is applied to any sound signal, not indicating the content in any way.

Audio has been transmitted digitally over networks for several decades. In 1947, Black and Edson from Bell Telephone Laboratories described a method which transforms "voice wave into coded pulses" [10]. This method was named as "Pulse Code Modulation" (PCM). Since then, numerous new audio codecs have been introduced as digital technology and telecommunications in general have evolved.

In mobile telecommunications, minimization of the amount of bits required to transmit audio signal of a single caller is important in order to serve as many clients per base station as possible. This means that the coding algorithm has to compress the speech in a way that the encoded signal uses only a small amount of bits. An audio coding algorithm can be designed to be as efficient as possible when the type of audio content is known.

The most typical audio signal to be transmitted via mobile and other networks has been, and most probably will be, speech. This has led to a situation where audio codecs used in mobile telecommunications have traditionally been designed to be as efficient as possible with speech content. This is why these codecs are often referred to as speech or voice codecs. This kind of coding algorithms might not perform well with other types of audio, such as music or sound effects. Audio codecs are commonly (and in this thesis) classified in terms of sampling frequency as follows:

– Narrowband, 8kHz

- Wideband, 16khz and over
- Fullband, 40kHz and over

In order to understand the foundations of recent codec technology, some of the basic methods commonly used in speech and other audio coding are briefly reviewed in the following.

Linear Predictive Coding (LPC)

LPC was first introduced in early packet-based voice communication over ARPAnet in 1974. ARPAnet is known as the predecessor of the Internet. It could provide understandable speech with a low bitrate of 2400 bps [11]. The method has been developed further over the years but the basic idea behind it has remained as an important tool in speech coding at low bitrates.

The source in human speech production are the glottal pulses that are generated when air is pushed through the vibrating vocal chords. Speech is formed when this source signal is modified by the vocal tract. The vocal tract acts as an "acoustic tube" of variable size and shape. It produces different resonance peaks in the speech spectrum that are called formants. The source (glottal pulses) can be modeled using a train of impulses or random noise, depending on whether a voiced vowel, such as /a/ or /i/, or an unvoiced fricative such as /k/ or /s/ is concerned. Different resonances produced by the vocal tract can be approximated using digital filters with time-varying parameters. [12]

The assumptions described above lead to the conclusion that the mechanisms of human speech production can be approximated as a linear system, consisting of a source and a filter. The idea is to transmit the parameters that characterize the source and the filter instead of the actual speech samples. In order to utilize this kind of model in speech coding, one must have a method of determining these parameters. Naturally these parameters change over time and have to be constantly redetermined by analyzing the source signal. A common tool for this kind of analysis a mathematical operation called linear prediction.

The principle behind LPC is that a speech sample can be estimated as a linear

combination of the past samples. This is can be expressed as

$$s(n) = \sum_{k=1}^{p} \alpha_k s(n-k) + e(n) \tag{1}$$

where $s(n)$ is the current sample, p is the order of the prediction filter, $e(n)$ is the prediction error and α_k is the kth filter coefficient. The speech is analysed in small sections called frames. The length of these frames is typically 20 ms, which is considered to be enough for capturing the envelope of the formants. The coefficients α_k are determined by minimizing energy of the error between the filtered predicted speech frame and the original speech frame. The energy of the error signal can be written as

$$\sum_n e^2(n) = \sum_n \left(s(n) - \sum_{k=1}^{p} \alpha_k s(n-k) \right)^2 . \tag{2}$$

The minimum energy can be found by differentiating (2) with respect to α_k and solving the roots of the resulting p equations. These equations can be written in a matrix form and solved by for example using the Levinson-Durbin Recursion. The remaining error, often called the residual or the innovation, is small in amplitude and can be represented with a small amount of bits. The speech information can be transmitted efficiently by transmitting only the calculated FIR coefficients that represent the formants, and the residual signal. [13]

At the receiver, the formats are reconstructed frame by frame by filtering the prediction error (or random noise or an impulse train in a simple implementation) with a filter that is the inverse of the analysis filter. The analysis filter was used at the sender to calculate the prediction error. In other words, the source signal is modified so that its spectrum resembles the spectrum of the original speech frame. [13]

In a simple implementation, the choice of source signal in reconstructing a speech frame is done by determining whether the original frame of speech is voiced or not. For example, frames originating from vowels are voiced sequenced. At time domain they are periodic, meaning that they have a certain pitch. On the other hand, consonants produce unvoiced frames that don't have a distinct pitch. In the simplest case, an impulse train with a corresponding pitch can be used as source for voiced frames and random noise can be used for unvoiced frames. [13]

Perceptual audio coding

Perceptual audio coding refers to coding methods that utilize psychoacoustic phenomena in order to compress audio data. Uncompressed audio contains irrelevant signal information that is not detectable even by a well-trained listener because of certain properties of the human auditory system. Psychoacoustics is a field of research that studies these properties. Most of the common audio codecs that are designed for music rely on psychoacoustic principles in compression of audio data. MP3 and AAC are well known examples of this kind of audio coding. The two most important psychoacustic phenomenas utilized in audio compression are absolute threshold of hearing and masking effects. [14]

Absolute threshold of hearing refers to the amount of energy needed in a pure sinusoidal sound wave that it can be detected by a listener in a noiseless environment. This threshold depends on the frequency of the sinusoidal sound wave. The described phenomenon can be utilized in audio compression. This is done by shaping the quantization noise in a way that focuses its energy into frequencies that the human ear is less sensitive to. Quantization noise is a common artifact of audio compression that is typically caused by limiting the bit amount that is used to represent the audio signal. [14]

There are different kind of masking effects present in the human auditory system, such as frequency masking and different types of temporal masking. Frequency masking is widely utilized in audio compression. The basic method for this is to remove redundant signal components that cannot be heard because of the masking effect. Frequency masking means loosely that a sound source can mask another sound source if their sound energies are focused at similiar frequencies. The human auditory system performs spectral analysis in the cochlea (inner ear), along the basilar membrane. [14]

The basilar membrane reacts to mechanical vibrations in the cochlea. Different frequency components of a sound wave can be perceived because they create resonances at different locations on the basilar membrane. Neural receptors are located around the basilar membrane and they send impulses to the brains whenever a matching frequency component is present. Because the basilar membrane reacts on a slightly larger area than just at the point of resonance, neighboring receptors get activated also. This is why a weak frequency component resonating at the point of

the neigboring receptor will not be perceived. This kind of frequency components are redundant and can therefore be removed. [14]

Modified Discrete Cosine Transform (MDCT)

In order to process the frequency components of audio signals directly, a transform is required. Here, a transform refers to the process of mathematically moving from one representation to another (in this case, from time-domain to frequency-domain). MDCT is a widely used tool for this purpose in audio coding. It is used in various popular codecs, such as the aforementioned MP3 and AAC. It solves many problems encountered when using other transform-based methods. [15]

For a transform-based coding method to be efficient, it needs to be critically sampled. In this context, it means that the total number of samples in one domain needs to equal the number of samples in another. Critical sampling as such could be achieved with other methods than MDCT, such as Discrete Fourier Transform (DFT) with rectangular, non-overlapping windowing. Unfortunately, rectangular windowing is undesirable because of the discontinuities that occur at the edges of the window. These discontinuities cause errors in the transform. This problem can be solved by using a tapered window in sampling and by overlapping the windows in a way that maintains the original signal power. Then again, this overlap-add -method causes the number of samples to increase and thus the signal is not critically sampled. [15]

MDCT solves this problem by utilizing time domain alias cancellation (TDAC). MDCT uses 50% overlapping but the transform itself produces only half of the input samples as output. MDCT is given as

$$\alpha_r = \sum_{k=0}^{2N-1} \tilde{a}_k \cos\left[\pi \frac{(k + (N+1)/2)(r + 1/2)}{N}\right] \quad , r = 0, ..., N-1 \tag{3}$$

where $\tilde{a}_k$ is the windowed input signal [15].

When the transform is inversed, the number of samples in the frequency domain is only half of the number of samples in the resulting time domain signal. Inverse MDCT is given as

$$\hat{a}_k = \frac{2}{N} \sum_{r=0}^{N-1} \alpha_r \cos\left[\pi \frac{(k + (N+1)/2)(r + 1/2)}{N}\right] \quad , k = 0, ..., 2N-1 \tag{4}$$

where $\hat{a}_k$ is the time-domain signal containing aliased components. These aliased components occur in adjacent overlapping frames as equal in magnitude but opposite in phase. This means that the aliased components can be cancelled by summing the overlapping parts together. Now, the process is shown to be critically sampled and to offer the benefits of overlap-add. Figure 4 illustrates how the TDAC is achieved in MDCT. [15]

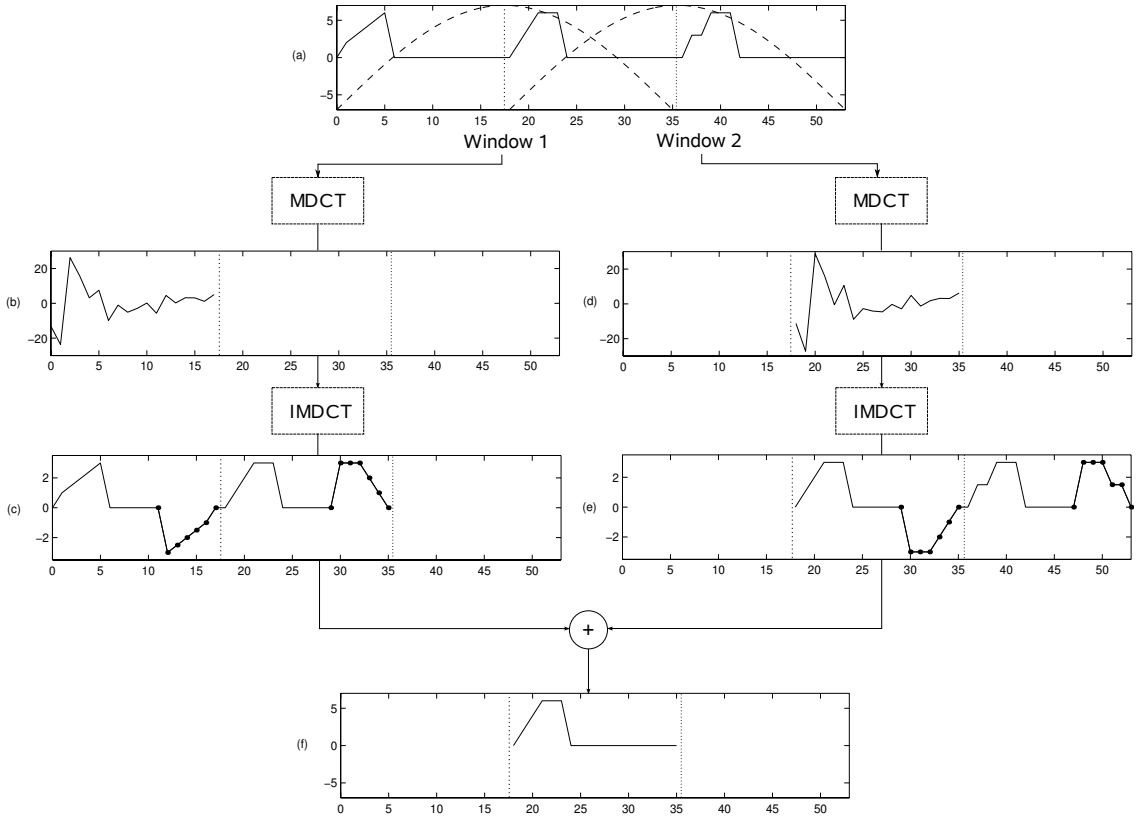

Figure 4: Principle of TDAC in MDCT. Starting from the top, an arbitrary time domain signal and sinusoidal windows with 50% overlaps are shown in (a). MDCT's of the adjacent windows are plotted in (b) and (d). The results of the inverse transforms are shown in (c) and (e) with the alias components marked with dotted lines. The result of the overlap-add is plotted in (f), showing the reconstructed middle part from the original time-domain signal in (a). Picture adapted with permission from [15].

Entropy coding

The final stage of preparing the audio (and video) data to be transmitted is known as entropy coding. The purpose of entropy coding is to minimize the amount of bits

required in transmission. This part of the coding process is lossless, meaning that it does not introduce any extra distortion into the transmitted audio signal [16]. A widely used method for entropy coding is known as Huffman coding [17].

The idea in Huffman coding is that instead of transmitting the actual binary data, codewords are transmitted. These codewords are selected in a way that prevents any single codeword from being confused with another. This means that the only allowed codewords are such that do not appear digit for digit in other codewords of greater length. Because of this, codewords can be concatenated before transmission without any separating characters such as commas or semicolons. Maximum efficiency is achieved in Huffman-coded transmission when the propability of a certain datapoint is known. This way the shortest codewords can be assigned for the most common datapoints and to total amount of bits needed in the transmission is minimized. [17]

2.2.2 Video coding

In 1993, the growing customer demand for video conferencing services led to the first recommendation for compressing video data for transmission over a network [18]. As for the audio codecs, numerous new video codecs have emerged since and their properties have been evolving towards a higher quality and higher compression rates. In telecommunications, general increase of bandwidths and lowered costs of processing power and memory have enabled the transmission of HD-grade (High Definition) video streams over IP networks [19].

Digital video consists of adjacent images. Each image corresponds a single frame of data. The rate of changing the individual images is denoted as frames per second (FPS). Video coding uses similiar compression methods as audio coding. Whereas audio signals contain only temporal data, video signal contain temporal and spatial data. Redundancy can be minimized from both of these dimensions. Prediction is commonly used instead of transmitting each frame pixel by pixel. Individual frames are often divided into blocks of for example 8x8 pixels. If there is no difference between adjacent frames in an individual block, the given block does not need to be transmitted. If there is a difference, a neighboring block that matches best is found. The difference between these blocks can be used for further compression. [20]

Instead of MDCT, a commonly used transform method in video coding is the basic DCT. DCT is performed typically on the 8x8 pixel frame block. In this case, it

yields 64 frequency-domain coefficients. The human eye is most sensitive to the image components that are represented in low frequency components of the transformed image block. Low frequencies in image data correspond to large features in an image, whereas high frequencies correspond to small features. This is utilized in compression algorithms by assigning more bits to low frequencies which are perceptually more significant. [21]

2.3 Multimedia codecs in telecommunications

Some common multimedia codecs that are used in telecommunications, and their main properties are listed below in tables 1 and 2. In the following, codecs that are used in the upcoming case study in Chapter 4 are briefly reviewed. A more detailed discussion of two recent hybrid audio codecs will follow in Section 3. Detailed discussion on video codecs is out of the scope of this thesis.

Table 1: Examples of common audio codecs in telecommunications [22] [23] [24] [25] [26] [27] [28].

Codec name	Year of approval	Sampling rate	Bit depth	Compressed bitrate
G.711	1972	8 kHz	8 bit	64 kbit/s
G.722	1988	16 kHz	14 bit	48-64 kbit/s
EFR	1995	8 kHz	13 bit	12.2 kbit/s
AMR-NB	1999	8 kHz	13 bit	1.8-12.2 kbit/s
AMR-WB	2001	16 kHz	14 bit	6.6-23.85 kbit/s
G.729	2007	8 kHz	16 bit	8 kbit/s
G.719	2008	48 kHz	16 bit	32-128 kbit/s

Table 2: Common video codecs in telecommunications [29] [30].

Codec name	Year of approval	Supported resolutions
H.263	1996	128 X 96
		176 X 144
		352 X 288
		704 X 576
		1408 X 1152
H.264	2003	128 X 96
		.
		. (Several)
		.
		4096 X 2304

2.3.1 G.711

G.711 is the basic speech codec that has been used since the digitalization of fixed-line telephone calls. It was published in 1972 by ITU-T as "Pulse Code Modulation of Voice Frequencies" and it is also commonly referred to as PCM. G.711 does not use any compression algorithms, instead it compands the signal by distributing the available dynamic range unevenly. [22]

There are two versions of the codec, G.711 A-law and G.711 μ-law. Both versions use a non-uniform scale to represent individual speech samples. The difference between the two is in the shape of the input-output relation curves. G.711 μ-law assigns more bits to lower level signals than G.711 A-law. This means that A-law offers a wider dynamic range whereas μ-law provides a better quality for low level signals. G.711 μ-law is the accepted standard in United States and Japan whereas G.711 A-law is the accepted standard in for example Europe. [31]

2.3.2 Adaptive Multirate Narrowband (AMR)

AMR is used in 3G mobile networks as a basic mandatory speech codec. AMR is a speech codec that includes eight different modes of operation that encode the audio signal in different bitrates, ranging from 4.75 to 12.2 kbps. The bitrate is chosen by continuously monitoring the quality of the available radio link between the terminal and the base station. When the radio link conditions are good, a high bitrate option is used for best available speech quality. On the other hand, when the radio link is poor, a lower bitrate in the source coder is used and more bits are allocated for error resilience in the transport coder. AMR consists of four different functions, including speech compression, voice activity detection, comfort noise generation and error concealment. [32]

Speech compression in AMR is based on code-excited linear prediction (CELP). Code excited in this context means that the source signal used in synthesis is selected from a table of pre-determined options. The suitable source signal is selected at the sender using analysis-by-synthesis method and the selection is coded within the transmitted frame of speech. [33]

AMR has been extended to support higher sampling rates and to handle general audio signals better. These extension are named as AMR wideband (AMR-WB) and Extended AMR-WB (AMR-WB+). AMR-WB+ includes a hybrid mode of

operation that combines compression techniques from speech and music codecs. Since the extensions, the basic version of AMR has since been referred to as AMR narrowband (AMR-NB). [34]

2.3.3 H.263

H.263 is a video codec that was originally designed for low bitrate communication applications such as early forms of video conferencing. H.263 was approved by ITU-T in 1993 and it is based on its predecessor H.261. Video compression in H.263 is mainly based on using inter-picture prediction for removing temporal redundancy and DCT in removing spatial redundancy. [30]

H.263 has been extended twice since its first approval to support a larger variety of resolutions. In addition, the available features and coding tools of the codec were divided to a number of profiles and levels. These extension were officially published as Annexes I-W and Annex X, and they were combined into a unified document in 2005.

The current version of H.263 includes these extensions [35]. The extended versions of H.263 are informally known as H.263+ and H.263++. In the specification of IMS, H.263 (profiles 0 and 3 level 45) is a mandatory video codec [36].

2.3.4 H.264

H.264 is a widely used video codec that was approved by ITU-T in 2003 as a successor to H.263. From the point of view of telecommunications, H.264 was designed to be a flexible codec that can be used in a wide variety of network environments. It was developed as a response to the growing need for higher compression of video than the H.263 could achieve. This higher compression is computationally demanding and has been made possible by the general increase of processing power and advances in video compression tehnology. Video compression in H.264 is based on similiar but further developed methods as in H.263. [9]

The subsets of features or coding tools of the codec have been divided into a number of profiles and levels. A level indicates the memory and computational requirements for a given set of features. A certain profile is usually targeted for a family of applications that are similiar in terms of requirements for memory, processing, latency and error resilience. [9]

Baseline profile is targeted for applications that require low latencies and high error resilience, such as video conferencing. High profile is suited for playback from a mass storage and for broadcast applications. It excludes the error resilience tools and introduces more latency because of increased computational costs, but results in a better picture quality. Various other profiles are also defined that offer options for different transmission conditions. In the specification of IMS, H.264 (baseline profile, level 1b) is a mandatory video codec [36]. [9]

2.4 Microprocessors in telecommunications

A microprocessor, commonly referred to as a processor, is a device in a computer system that executes operations formed by sequential or parallel instructions. Instructions are small primitive tasks that are built in a processor. Instructions are organized in the processor's memory based on assembly-code that defines the program that the processor will run. Assembly-code can be written by a programmer directly or it can be produced by a compiler program from a higher-level programming language such as C,C++ or Java.

In telecommunications, certain applications or functions have traditionally been handled by specific types of microprocessors. General purpose processors (GPP) have served as network controllers and as hosts for operators managing tools. Digital signal processor (DSP) is a type of microprocessor that is best suited for multimedia processing such encoding and decoding. Network processors are used for performing common tasks in data transmissions. This improves the throughput of data packets in real time communications. [37]

The performance of a processor depends on many aspects. One of the key factors is the processors clock speed, and for many years it was considered as the most important measure of performance. Clock speed determines the speed of executing single operations. Another important factor is the cache memory, which is a relatively small but very fast memory that is embedded in a processor. The purpose of a cache memory is to provide a fast access to common operations executed by the processor. The maximum size of a cache memory depends on the amount of primitive memory elements that can be fitted in the processor core along other components. In a typical microprocessor, this primitive element is a transistor.

Moore's law states that the maximum number of transistors on an integrated circuit will double in approximately two years. While the maximum practical clock speeds

have been reached in microprocessors due to physical limits of current semiconductor technology, Moore's law still seems to apply. This means that the amount of transistors in a given area of silicon, the material from which integrated circuits are made of, can be increased. In addition to being able to increase the number of transistors on a single processor core, the number of cores in a single processor can be increased. A core is a single processing unit, and there can be several cores embedded in a single processor. This kind of a processor is called a multicore processor, and it includes two or more cores embedded in one physical piece of silicon, commonly referred to as a chip.

Multi-core chips can be divided into heterogeneous and homogeneous devices. Homogeneous devices contain multiple instances of identical processing cores. Computational load of the applications is divided between these cores and processing tasks can be run in parallel, decreasing the total time of execution for a given set of tasks. A heterogeneous device contains different types of processing cores and co-processors within a common silicon chip. Co-processors often serve as hardware implementations of some common algorithms, such as H.264 video coding. This kind of a heterogeneous device is also often referred to as a System-on-Chip (SoC). [37]

In some tasks, using multicore processors makes it possible to do divide the processing to separate cores which will decrease the required computation time of the tasks. This kind of tasks typically involve a large amount of data that has to be processed in a given time window. The general demand for data processing capacity in telecommunications, and also in most other industry areas is continuously increasing. This has caused several microprocessor vendors to start migrating all the different processor types towards multicore architectures.

3 Hybrid audio codecs – recent advances in standardization

As the means of communicating over networks are converging, the need for more general audio coding scheme that could handle all kinds of audio material efficiently and with good quality has emerged. The purpose of this chapter is to review two new hybrid audio codecs, assessing their properties and benefits on a high level, based on available literature.

3.1 Universal Speech and Audio Coding

A new standard for low-bitrate audio coding was published in 2012 as "MPEG-D – Unified Speech and Audio Coding" (USAC) by the MPEG Audio Subgroup. The purpose of this new codec is to bring together speech and general audio coding, which have traditionally been treated separately. It combines elements from both areas in order to provide a unified codec solution which is meant to work with all audio content, regardless of whether it originates from speech, music or any other type of sound. USAC was originally developed jointly by Fraunhofer IIS and VoiceAge Corporation as "Extended HE-AAC". [38]

Description

USAC utilizes coding methods that are adapted from existing speech and music codecs. In addition, some new methods have been developed in order to make the adapted methods work together. USAC is based on an earlier audio codec called High Efficiency Advanced Audio Coding version 2 (HE-AACv2), and it preserves the same overall structure. Nevertheless, USAC implementations are not required to be backwards compatible with AAC or HE-AAC.

The structure of the USAC encoder is illustrated in Figure 5, and an overview of the USAC decoder is shown in Figure 6. A complete description of the functional blocks can be found in [39]. Some of the coding methods used in USAC are briefly reviewed in the following, including frequency domain coding (FD), transform coded excitation (TCX), algebraic code-excited linear prediction (ACELP) and forward aliasing cancellation (FAC). These have been mentioned in [40] as the parts that

give the greatest increase in performance compared to other solutions when coding mixed audio content at low bitrates.

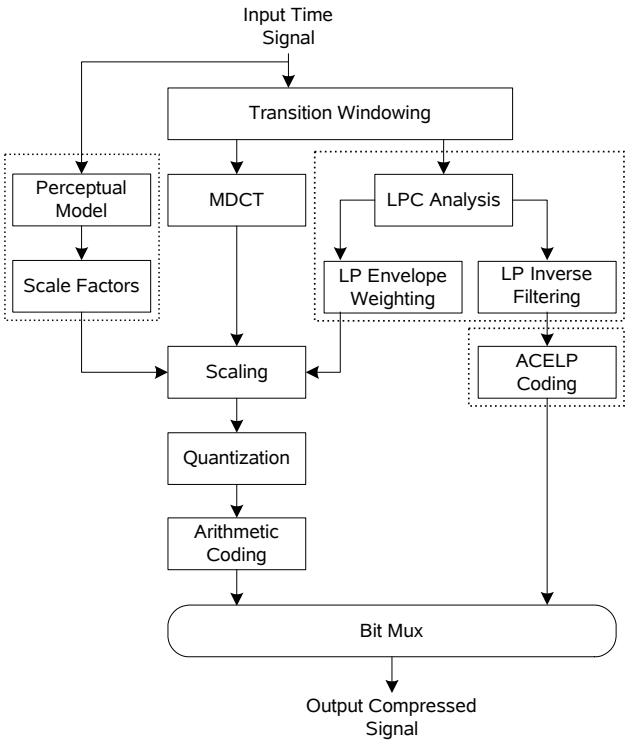

Figure 5: Structure of the USAC encoder. Parts that are separated by dotted lines have remained as separate blocks from adopted coding methods. Adapted from [40].

In USAC, FD refers to the classic perceptual coding parts that have been adopted from AAC. In FD, audioframes are first processed with MDCT (discussed in section 2.2.1). The transformed input frames are then scaled based on a perceptual model that emulates the masking phenomena of human hearing. [39]

TCX has previously been used as a speech coding method, for example in AMR-WB+. It uses short-term linear prediction (LPC) that has been extended to transform the residual signal into frequency domain using MDCT. Whereas FD uses a perceptual model, TCX uses an explicit source model. In other words, it can loosely be said that TCX models the human speech production, whereas FD models the human auditory system. TCX is not as accurate in terms of frequency resolution as FD, but it is simpler and faster and produces a fairly good representation of a speech signal. USAC uses either of these coding methods to code an audioframe. The selection of coding method is done by a separate logic based on content recognition of the input signal. TCX is typically selected for speech and for FD for music. [39]

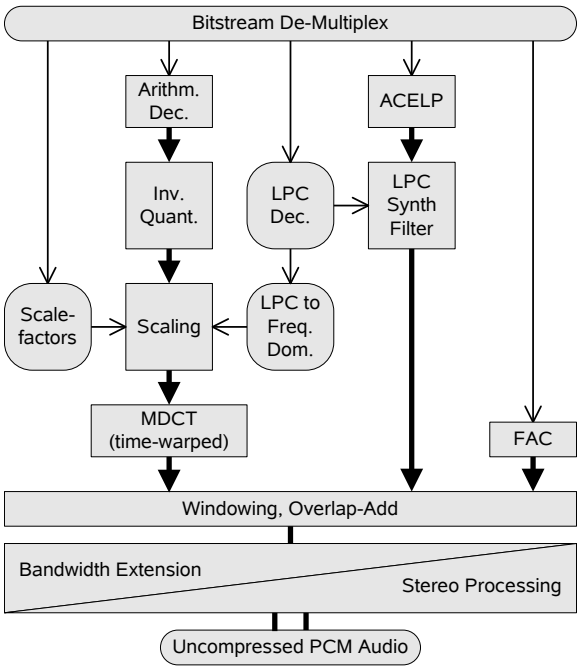

Figure 6: Structure of the USAC decoder. Adopted from [39].

ACELP is used in USAC to further improve the coding performance for speech signals. ACELP uses the same short-term linear prediction filter as the TCX to model the vocal tract. Additionally, it incorporates a long-term prediction filter that models the periodic excitation signal produced by the human vocal chords. Audioframes that are fed into ACELP are divided into four subframes. These subframes are then divided again into four parts to achieve a very high time resolution compared to FD and TCX. This allows ACELP to recognize attacks of sound events accurately. [39]

In decoding (synthesis), frames encoded with ACELP are windowed with square windows and with no overlapping with adjacent frames. On the other hand, sinusoidal windowing with overlapping has to be used with frames encoded with MDCT-based method in order to achieve TDAC. In USAC, TDAC only happens automatically when the adjacent frames are encoded with MDCT-based method. In some situations, adjacent frames might also be encoded with different methods. For example, one frame could be ACELP-coded and the following could be FD- or TCX-coded. In this kind of situation, TDAC will not happen without additional processing. The solution to this problem in USAC is the use of FAC. [39]

FAC is a method where parts of the signal that are necessary for cancelling time domain aliasing are synthesized. These parts have to be created at the encoder. The

encoder must determine what has to be added to a frame encoded with ACELP in order to completely remove time domain aliasing in the decoder. This FAC-signal must then be encoded and transmitted in order add the alias-cancelling components into the decoded audioframe at the receiver. [39]

Performance

Results of a sound quality evaluation of USAC were reported in [40]. The test was carried out using the MUSHRA-methology (explained for example in [41]) and comparative test were made for HE-AACv2 and AMR-WB+. Codecs were tested using various bitrates. Mono signals were tested with bitrates ranging from 12 kbps to 24 kbps, and stereo signals with bitrates ranging from 16 kbps to 64 kbps. Test signals consisted of speech, music and mixed content. Results showed that USAC performs at least as well or better than HE-AACv2 and AMR-WB+ at all tested bitrates and types of audio content types. When results are averaged over all three content types, USAC achieved the highest points. The difference in MUSHRA mean scores between USAC and AMR-WB+ was reported to be approximately five points in favor of USAC for mono signals. For stereo signals, MUSHRA means scores of USAC and HE-AACv2 converged towards higher bitrates but at 16 kbps, USAC scored approximately ten points more. Similiar test results were reported in [39].

Conclusion

USAC is a recent audio codec which has performed very well in terms of percieved sound quality in the reported tests [39] [40]. It is an MPEG standard, meaning that the codec is under licensing, and therefore non-free. Exact measures of the required latency in USAC were not found by the thesis writer, but because the codec is widely based on HE-AAC-v2, it is likely that USAC introduces a fairly high latency in the coding process and therefore is not well suited for real-time applications.

3.2 Opus

A working group created by the Internet Engineering Task Force (IETF) is working on a fullband audio codec that is meant to work with general audio material, including speech and music. The codec is named as "Opus", and the development was in

the final stages of preparing the first production release at the time of writing this thesis. Opus combines technologies from Skype, Octasic, Xiph.org and Broadcom. When released, Opus will be an open source codec. [42]

Description

Opus is based on two existing codecs that have previously been used separately. These are known as SILK and Constrained-Energy Lapped Transform (CELT). Opus can be used in three different modes which are based on either or both of them. Combination of the two codecs is called the hybrid mode. In hybrid mode, Opus divides the source audio material into two frequency bands and codes them in parallel, using methods from SILK to code the lower band (up to 8 kHz) and methods from CELT to code the higher frequencies (8-20kHz). These two frequency bands are summed back together in the decoding phase. This is different compared to USAC, which uses content recognition to choose which coding method to use for a given frame of audio. [42]

SILK was originally a speech codec that was used in Skype's VoIP calls. Speech coding in SILK is based on linear prediction and the part of Opus that is SILK-based is thus called the linear prediction layer. Opus is not backwards compatible with SILK because of the heavy modifications made in the linear prediction layer compared to the original SILK. CELT is based on MDCT, and the part of Opus that is based on CELT is thus named as the MDCT Layer. [42]

Opus supports both fixed and variable bitrates, ranging from 6 kb/s to 510 kb/s. There can be either mono or stereo frames within a single stream. Frame length can adjusted in the range of 2.5 ms to 60 ms. In general, shorter frames require higher bitrates for equivalent sound quality. A block diagram of the Opus encoder is given in Figure 7. [42]

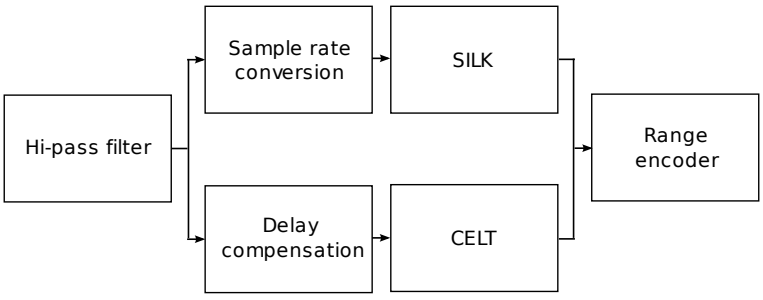

Figure 7: Structure of the Opus encoder. Adopted from [43].

Performance

Subjective sound quality of the Opus codec has been evaluated by the Nokia Research Center in 2011. Complete test description can be found in [44]. Opus was compared against various codecs, including AMR-NB, AMR-WB, G718B, G.722.1C and G.719. Method of the test was mean opinion score (MOS, described in Section 4.2) with an extended scale, ranging from 1 to 9. It was found that the LP mode of Opus provides a useable voice quality at low bitrates. In the listening test, the quality of speech was found to be slightly weaker than with AMR-WB. At moderate bitrates (20 kb/s - 40 kb/s), the hybrid mode of Opus was found to provide a higher quality than the other tested codecs. At higher bitrates, the MDCT mode of Opus shown to provide the best quality, although with more computational complexity. [44]

A similiar test was conducted by Jan Skoglund from Google in 2011. The test showed similiar results with medium bitrates when compared to the tests in [44]. Opus was found to provide better quality for wideband speech at 20 kb/s than AMR-WB at equal bitrate or G.719 at 32 kb/s. Opus showed a better quality in fullband stereo music at 64 kb/s than G.719 at equal bitrate.

The quality of the Opus codec with stereo and binaural content was tested in Universitaet Tuebingen in April 2011. Test showed that Opus performed well with binaural content at 64 kb/s, getting the highest MUSHRA scores of the tested codecs. With 32 kb/s, the tested version of Opus was outperformed clearly by AMR-WB+. [43]

Conclusion

IETF Opus is an open source codec that is under final preparations before the first release version. Its purpose is to act as an all-round codec for applications that require low latencies. Opus has performed well in listening tests with speech and music at medium bitrates and offers high quality at higher bitrates, although with a relatively high computational cost.

4 Case study: Evaluation of Tilera TILEPro64 -multicore platform in multimedia coding algorithms

4.1 Introduction and motivation

In a typical telecommunications application, the complexity and amount of coding algorithms sets the demands for the microprocessors. The introduction of HD-grade video services in telecommunications networks requires more complex coding algorithms that have increased the computational demands of the devices involved. It is typical that in a network node, there are several simultaneous coding processes running. Therefore, the amount of required processing in a network node becomes multiplied by the number of simultaneous codings.

Network nodes are typically real-time systems that process large amounts of data in strict time windows. As discussed in Section 2.4, this kind of processing tasks can benefit from using multicore processors. Therefore, for a company operating within network nodes such as Ericsson Finland, multicore processors form a relevant field of technology and it is important to follow the evolution of the field.

When the whole development process in concerned, it is not said that using multicore platforms always results in increased overall benefit. Traditionally, programming has been a linear process that can be thought as a sequiental series of events. When using a multicore platform, part of the operations are executed in parallel. This has to taken into account in the programming phase also. The programmer must have a way of assigning certain operations to certain cores, and this distribution of tasks is one of the key parts in developing applications for multicore platforms. The effort needed to handle the distribution of tasks is an important factor when concerning development efficiency. [45]

Tilera Corporation is a relatively new vendor of general-purpose multicore platforms. In this case study, Tilera TILEPro64 general purpose multicore platform (described in Section 5.1) was evaluated in terms of performance in multimedia coding, and the quality and usability of its development tools. The main purposes of the evaluation were to:

- verify the multimedia coding performance figures given by the vendor of the multicore platform.
- evaluate the quality and usability of the multicore development tools from a perspective of a developer who is not an expert in multicore programming.

Performance of the coding process does not only depend on raw computational power of the hardware because the efficiency of the codec implementation in software is important as well. This means that the tests performed for this case study also concern the specific codec implementations.

4.2 Test methods

Multimedia coding performance of the the evaluated multicore platform was tested by measuring the average time required to encode or decode a certain amount of media frames. Measurements were made in software, based on spent CPU cycles. Based on this measurement, the coding performance was calculated in frames per second (FPS). For video, a frame refers to an individual picture within the video stream. For audio, length of a single frame depends on the codec. In this case, a frame length of 160 samples was used. This corresponds to 20 ms of audio at a sampling rate of 8 kHz.

The encoding/decoding process of multimedia streams was verified informally by subjectively inspecting the end results. In addition, quality of the voice codec implementations was verified using perceptual speech quality -algorithm (PESQ) [46]. PESQ compares original and degraded audio files and gives an estimate of the impairment. Degraded audio file means that the audio data in the file has been encoded and decoded with a given codec. PESQ algorithm calculates a raw result that can be mathematically scaled to approximate subjective mean opinion scores (MOS) that would be obtained by standardized listening tests [47]. The mapping function from raw PESQ results to MOS scores is given as

$$MOS_{LQO} = 0.999 + \frac{4.999 - 0.999}{1 + e^{-1.4945 \cdot PESQ_{raw} + 4.6607}}. \quad (5)$$

MOS is expressed as a number in the range of one to five. One represents the lowest possible perceived quality and five the highest possible perceived quality. The meaning of different MOS scores is explained in Table 3.

Table 3: Explanation of MOS range [47].

MOS	Quality	Impairment
5	Excellent	Imperceptible
4	Good	Perceptible but not annoying
3	Fair	Slightly annoying
2	Poor	Annoying
1	Bad	Very annoying

The results obtained by scaling the raw PESQ results with equation 5 are formally called MOS-LQO scores [48]. These results were compared to reference results given by ITU-T in [49].

The quality and usability of the multicore development tools were evaluated informally during the process of developing the applications for the performance test.

4.3 Test setup

Test hardware

The multicore processor used in the evaluation was Tilera TILEPro64, which includes 64 fully programmable general purpose cores. Hardware architecture of Tilera multicore processors is discussed in more detail in Section 5.1. The test setup consisted of a host PC running Linux (CentOS release 5.7) and a Tilera TILEPro64 -based evaluation board connected to the host PC via PCIe bus. Details of the test hardware are listed in Table 4.

Table 4: Details of the test hardware.

	Host PC	PCIe card
CPU	Intel Xeon W3530	TILEPro64
CPU number of cores	4	64
CPU clock speed	2.8 GHz	866 Mhz
Memory	6 GB	512 MB
Operating system	Linux CentOS 5.7	-
Programming IDE	Tilera MDE 3.0.1.125620	-

Data transfer between the host PC and the evaluation board is physically done via the PCIe connection. In software, the data stored in the host PC can be seen by the evaluation board as mounted Linux directories. From these directories, the

data can be loaded into the card's internal RAM-memory. A block diagram of the TILEPro64 evaluation board is given in Figure 8, and a photograph of the actual device is shown in Figure 9.

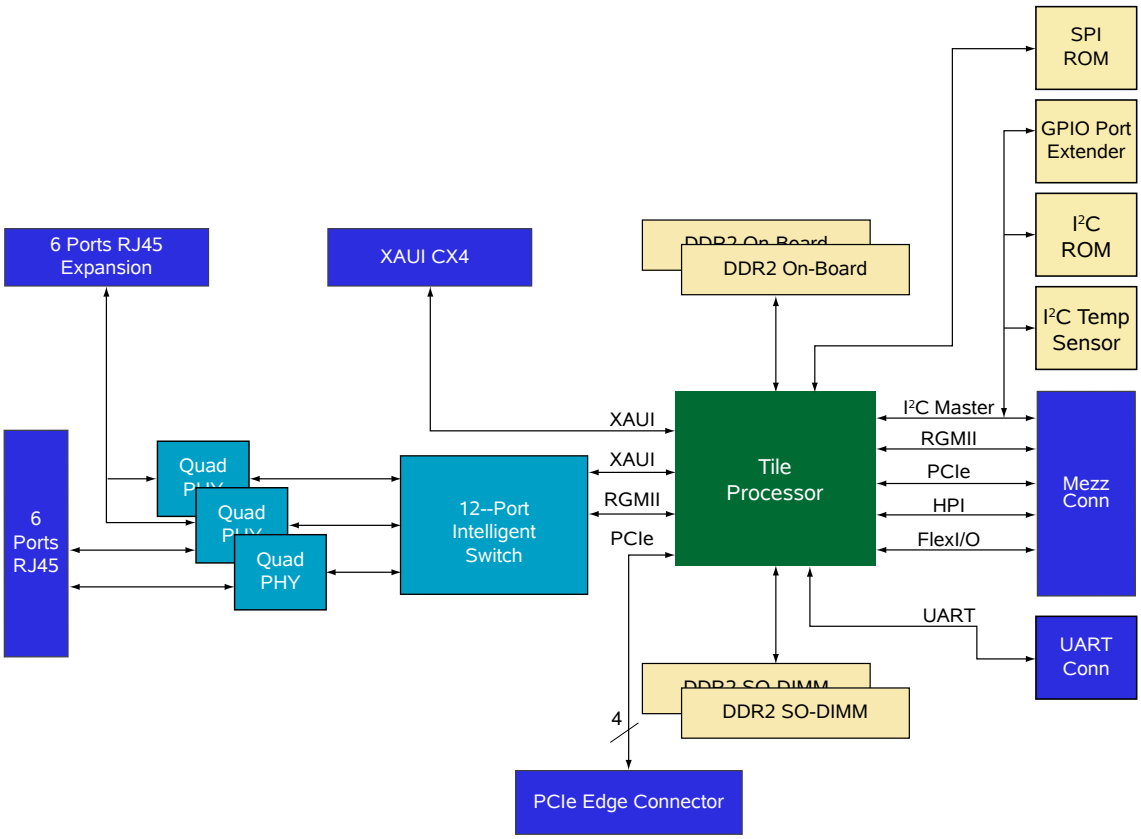

Figure 8: Block diagram of the Tilera TileExpress64 -evaluation card. The network connectivity of the card was not used in the tests. Adopted from [50], courtesy Tilera Corporation.

Test software

For this evaluation, Tilera provided an alpha2 version of a software library called Medialib that includes source code for multimedia codecs that have been optimized for TILE-series processors. Currently Medialib supports MPEG-1, MPEG-2, H.263, H.263+ and H.264 video codecs. Medialib also includes reference implementations of voice codecs, currently including G.711 (A-law and μ-law) and AMR-NB.

A set of test applications with source codes were also provided in the Medialib, and a set of custom test programs were developed for the performance tests based on these sample codes. Additionally, the quality of the available voice codec implementations

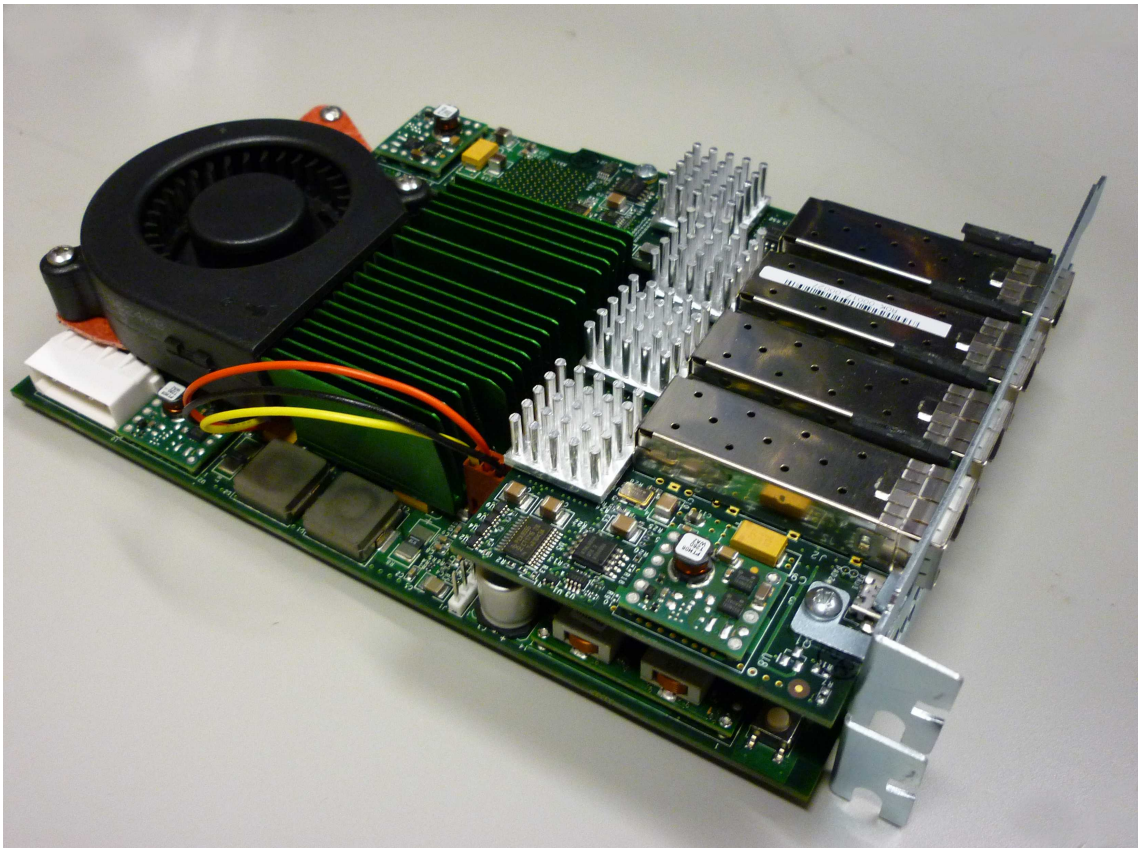

Figure 9: Photograph of the TILEPro64 -evaluation board.

was evaluated using the PESQ algorithm that was compiled from sources provided in [46].

The evaluation of the coding performance was done using two video codecs (H.264, H.263+) and three speech codecs (G.711 A-law, G.711 μ-law, AMR-NB) from the Medialib. For video codecs, two HD-grade resolutions were used (1280 X 720 and 1920 X 1080). H264 tests were run using the two profiles available for the tested implementation. These included constrained baseline profile and a limited version of high profile. The available features of high profile were CABAC and 8x8 transform support. More information on these profiles and methods can be found for example in [9].

Test files

Video streams that were used in the performace tests were provided as files with the Medialib library. The video files were stored in YUV 4:2:0 -format, which is a

common format for raw video files. There were two video files used in the tests for both tested resolutions. For audio coding tests, the test files used were provided by ITU-T in [51]. Original files were sampled at 16 kHz and for the PESQ analysis, they were downsampled to a sample rate of 8 kHz. This was done because the tested audio codecs use a sample rate of 8 kHz and the PESQ algorithm requires that the original and the degraded audio files are using the same sample rate. The test files included a set of speech sequences, of which four different speech files were used [51]. The selected files included recorded speech from two females and two males. Speech sequences included two sentences in American English from each speaker, for example:

"Pack the rackets in a neat thin case"

Details of the video test files are given in Table 5, and details of the audio files are given in Table 6.

Table 5: Details of the video files used in the tests.

File	Format	File size [MB]	Framerate [FPS]	Resolution
"Toys"	YUV 4:2:0	368	25	1920 X 1080
"Terrace"	YUV 4:2:0	534	60	1920 X 1080
"Conference"	YUV 4:2:0	790	60	1280 X 720
"Ships"	YUV 4:2:0	197	60	1280 X 720

Table 6: Details of the audio files used in the tests.

File	Format	Length [s]	File size [kB]	Sample rate [Hz]	Bit depth
"Male 1"	WAV	8	131.9	8000	16
"Male 2"	WAV	7	124.4	8000	16
"Female 1"	WAV	8	124.5	8000	16
"Female 2"	WAV	7	125.9	8000	16

4.4 Development of the test application

In order to evaluate the coding performance of the tested multicore platform, a software application was developed to run the encoding/decoding process and to measure execution times. The requirements for the test application in terms of functionality were fairly simple. Essentially it needed to be able to:

– Handle the input and output of multimedia files

- Control the amount of CPU cores used (later referred to as CPU pool utilization level)
- Initialize the coding algorithms
- Drive the encoding process
- Measure the duration of the process
- Provide performance results in FPS.

The sample source codes provided by Tilera implemented these functionalities. Therefore, there was no need for coding new functionalities. The development of the test applications was essentially a matter of modifying and combining the needed functionalities into a suitable form. The sample source codes were written in ANSI-C. Based on these, four applications were developed for the performance tests, EncodeVideo, DecodeVideo, EncodeAudio and DecodeAudio.

All of the developed applications share the same basic structure. First, the application reads a media file into memory. Then, the coding engine and runtime libraries are initialized. Finally, the program runs the particular coding tests several times using a predetermined amount of CPU cores at each run. The duration of each run is measured and the execution times are averaged over five runs at each CPU pool utilization level. In the final run, the output is witten into a file. In order to avoid delaying caused by disk I/O, the output is not written to disk as part of the actual coding performance tests. Performance of the disk I/O depends on the host PC and therefore is not related to the actual coding event. The application outputs the results in FPS in a textfile.

4.5 Test results

Results of the coding performance tests were plotted as a function of percentage of the available cores in use. This makes it possible to read the amount of required CPU capacity for a given frame rate. Using the default configuration of TILEPro64 evaluation board, the maximum amount of cores available was 57 cores out of 64.

Results of the video encoding performance tests are given in Figures 10 and 11. For H.264, the tests were run using constrained baseline profile (abbreviated BP in the figures) and high profile with limitations (abbreviated HP*, limitations described in

Section 4.3). The tests were run using a bitrate of 2 Mbps for 720p-resolution and a bitrate of 4 Mbps for 1080p-resolution. The decoding tests were run using the output data from the encoding tests as input data. The results are given in Figure 12 and Figure 13, respectively.

Audio performance was measured using input files from [46]. Results are given in FPS with a frame length of 160 samples. Test results for encoding and decoding audio are given in Figures 14 and 15.

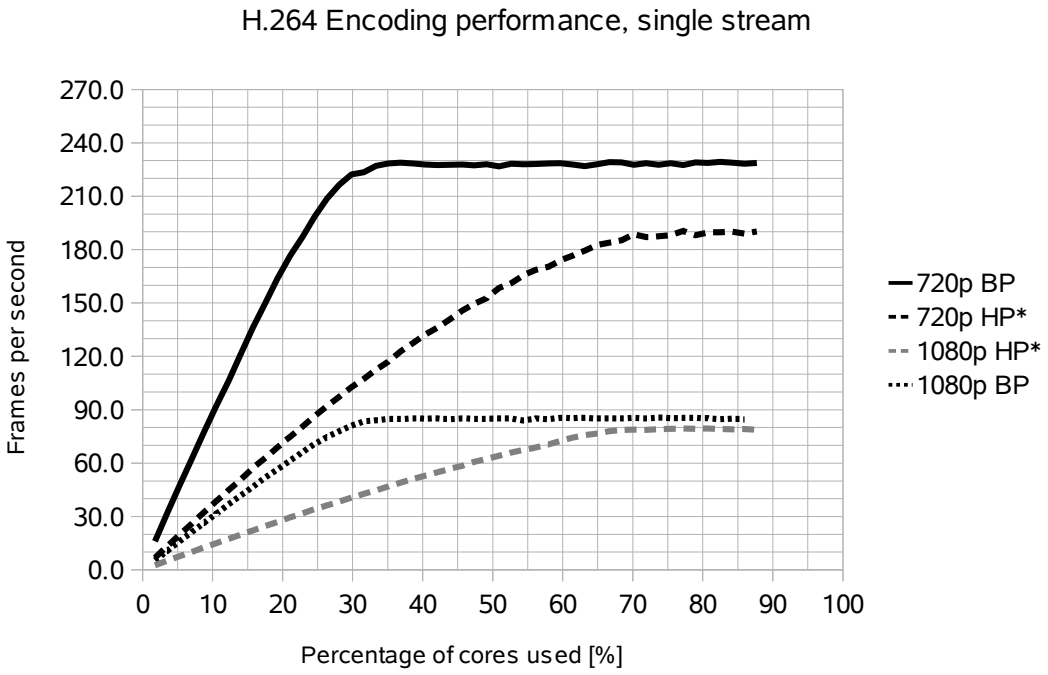

Figure 10: Average performance of the tested multicore platform in encoding H.264 video stream as a function of CPU usage.

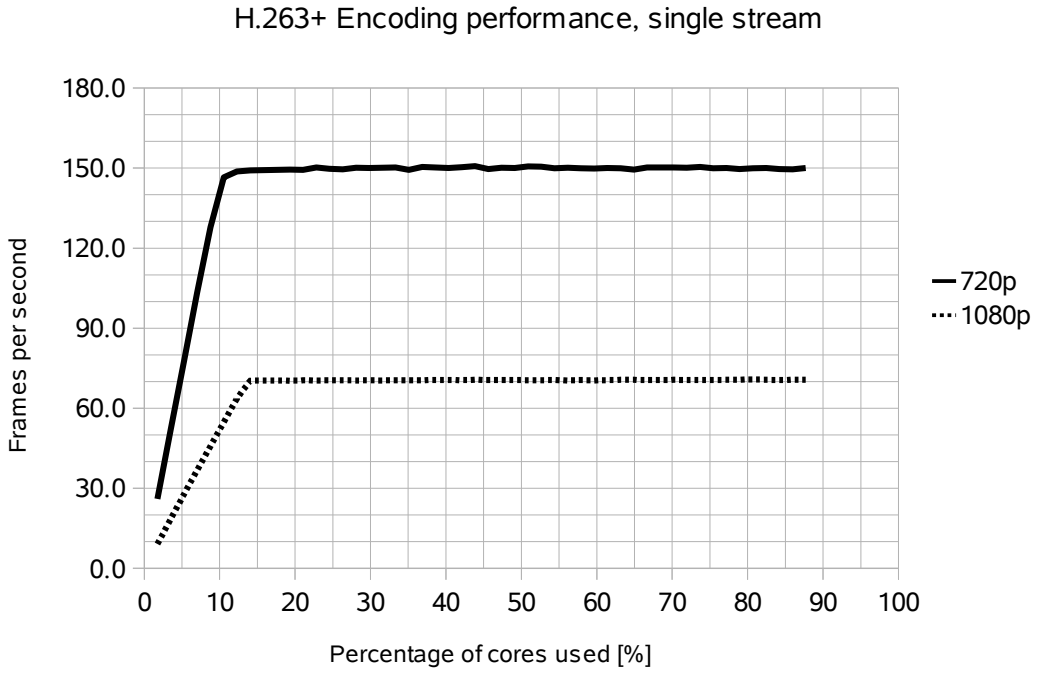

Figure 11: Average performance of the tested multicore platform in encoding H.263+ video stream as a function of CPU usage.

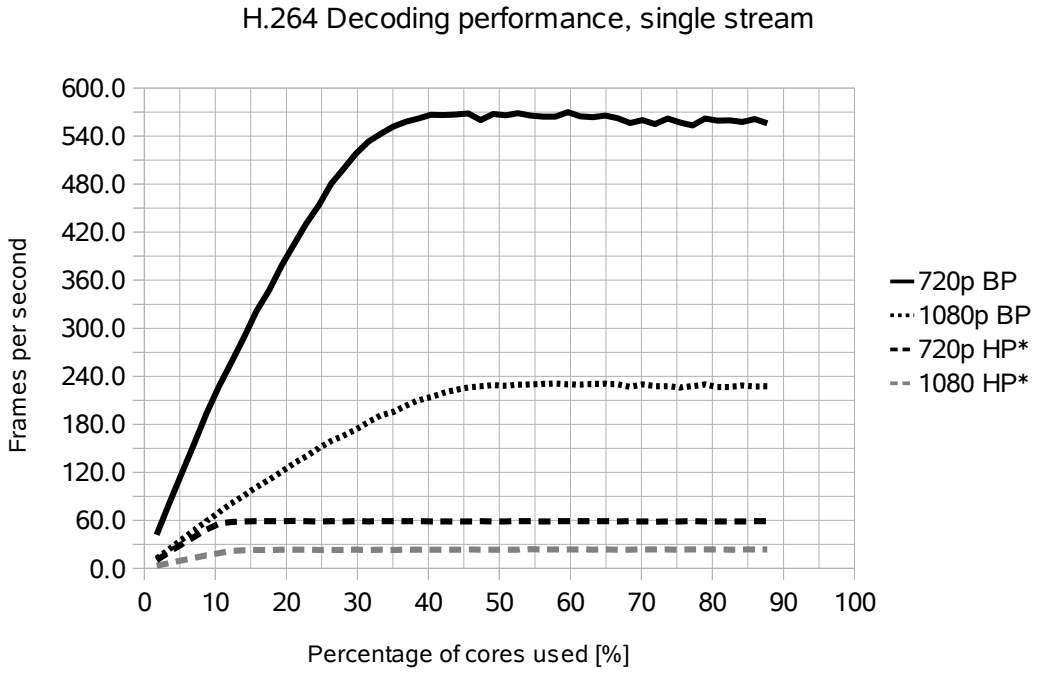

Figure 12: Average performance of the tested multicore platform in decoding H.264 video stream as a function of CPU usage.

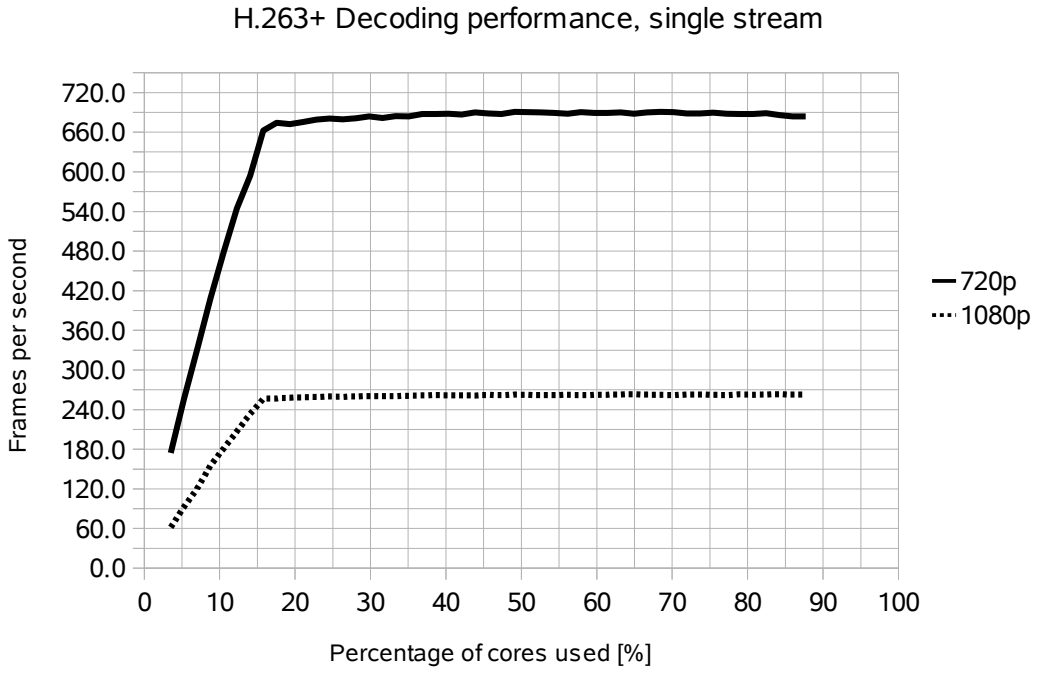

Figure 13: Average performance of the tested multicore platform in decoding H.263+ video stream as a function of CPU usage.

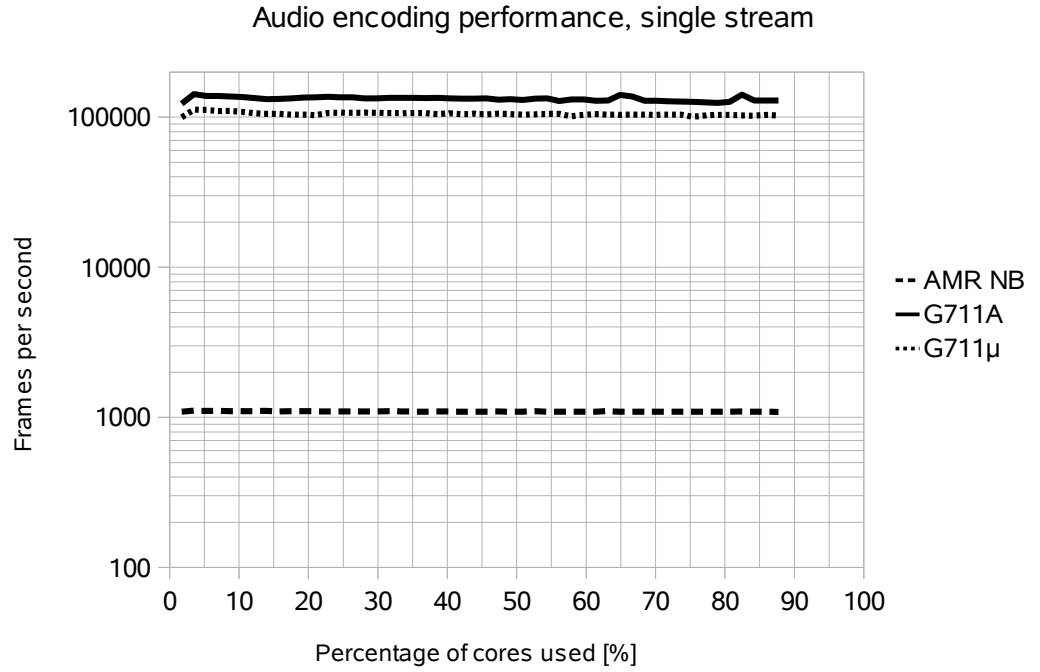

Figure 14: Average performance of the tested multicore platform in encoding audio stream as a function of CPU usage. Length of each frame is 20 ms.

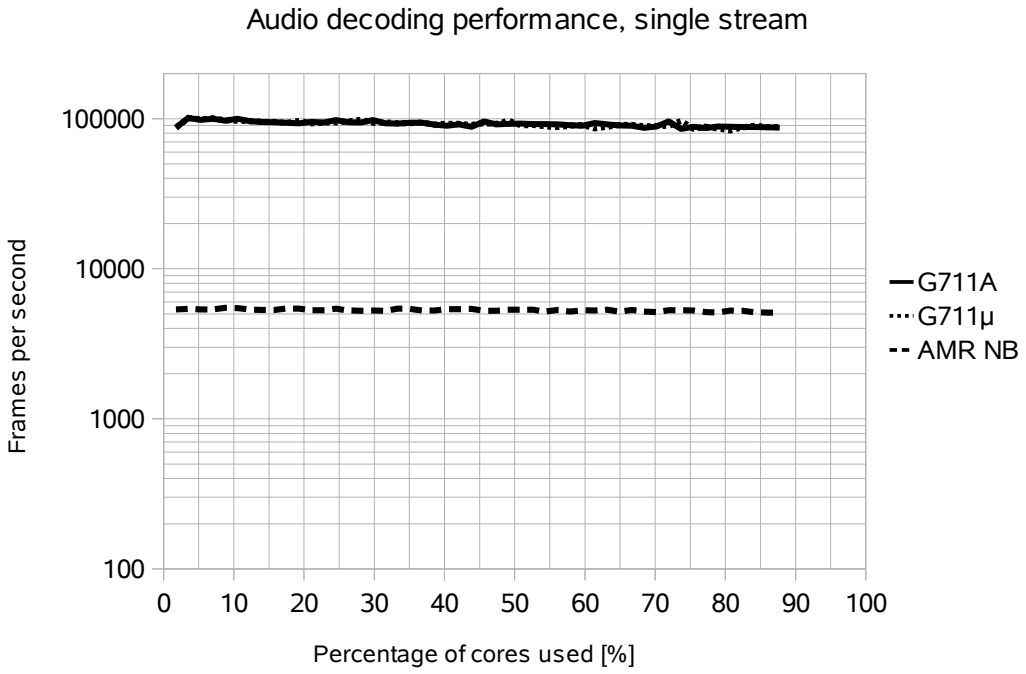

Figure 15: Average performance of the tested multicore platform in decoding audio stream as a function of CPU usage. Length of each frame is 20 ms.

4.6 Analysis

4.6.1 Performance tests

Figures 10-13 show that for video, the gained FPS results increase approximately in a linear fashion as a function of the used CPU capacity. At a point depending on the codec and profile, the FPS results saturate and increasing the amount of CPUs will not increase the gained FPS any further. Saturation points and the corresponding maximum FPS results for a single video stream for each tested video codec are shown in Table 7 for encoding and in Table 8 for decoding. Maximum FPS results were calculated by averaging the results beyond the saturation point.

Tilera's own measurements show that the saturation of the FPS results does not occur when multiple parallel video streams are used [53]. This implies that the hardware of the TILEPro64 does not cause the saturation. Instead, the amount of available parallelism for a single video stream is limited. Tilera's own measurement results for encoding eight simultaneous streams of H.264 video in 720p BP are shown in Figure 16.

Audio coding results in Figure 14 and Figure 15 show clearly that for narrow-

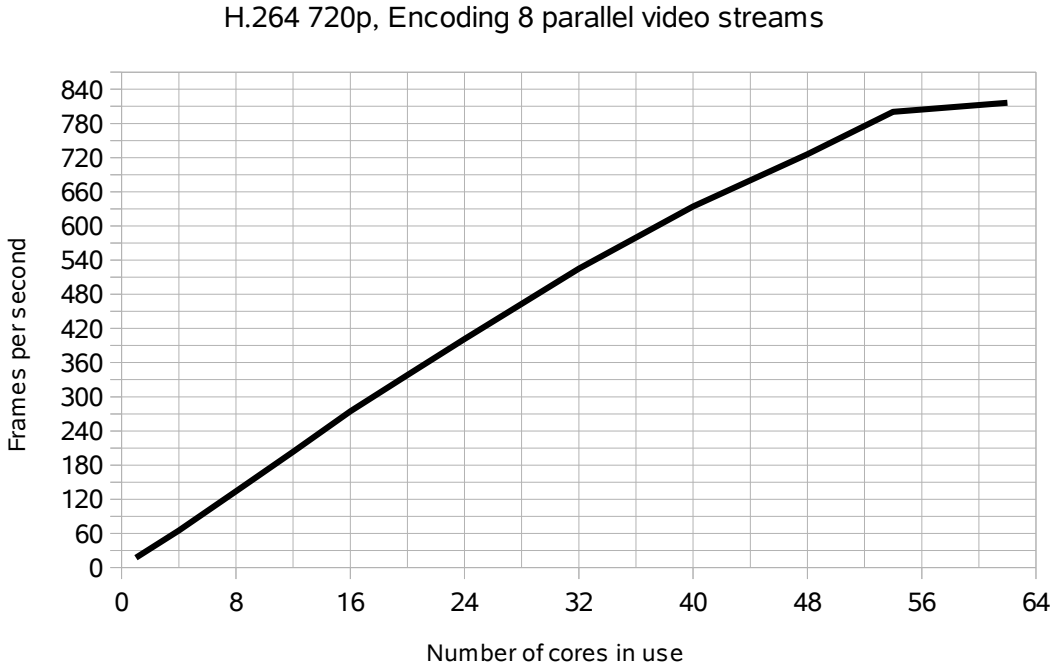

Figure 16: Tilera's measurement results for encoding eight simultaneous streams of H.264 video in 720p BP. Adapted from [53].

Table 7: Saturation characteristics of tested video codec implementations in encoding

Codec	Profile	Resolution	Saturation point	Max. FPS
H.264	BP	720p	35 %	228
	HP*	720p	70 %	188
	BP	1080p	35 %	86
	HP*	1080p	70 %	78
H.263+		720p	10 %	149
		1080p	15 %	70

Table 8: Saturation characteristics of tested video codec implementations in decoding.

Codec	Profile	Resolution	Saturation point	Max. FPS
H.264	BP	720p	40 %	563
	HP*	720p	10 %	59
	BP	1080p	50 %	229
	HP*	1080p	15 %	23
H.263+		720p	15 %	687
		1080p	15 %	70

band audio codecs, the tested codec implementations will not benefit significantly from using multiple CPUs for a single stream of audio. On the other hand, the computational cost of narrow band audio coding is low and the FPS results from a single CPU are already high.

FPS results of individual video files showed a relatively high variety of up to 30%. A larger and more diverse set of test data would give more reliable absolute performance results. Nevertheless, the trend of performance dependency on the amount of processor cores was similiar with all test files for a given resolution and codec profile.

4.6.2 Verification of the encoding/decoding process

The resulting video files of the encoding and decoding processes were informally observed as valid by inspectiong the results in VLC media player [54]. Encoded and decoded audio files were first observed as valid by listening through them informally in a media player software. Next, they were analyzed using PESQ-algorithm. Results of the analysis are discussed in the next section.

As an example of the effects audio encoding/decoding with Medialib audio codecs, a waveform and a spectrogram plot are shown in Figure 17 for the first sentence from test file "Male 2". The sentence spoken in this example is

"Both brothers wear the same size."

A waveform and a spectrogram plot of the same test file ("Male 2") after encoding to AMR-NB at 4.75 kbps and decoding back to a raw wave file is shown in Figure 18. As expected, after the encoding/decoding process the spectral components have slightly been changed. High frequency components and some formants have weakened, and background noise level has gone up across the frequency range. This is normal behavior with very low bit-rate codecs such as AMR-NB at 4.75 kbps.

4.6.3 PESQ analysis

The quality of the encoded audio clips was evaluated using the PESQ algorithm. The results showed that the percieved speech quality of the tested codec implementations was equal or better when compared with reference results given by ITU-T in [49].

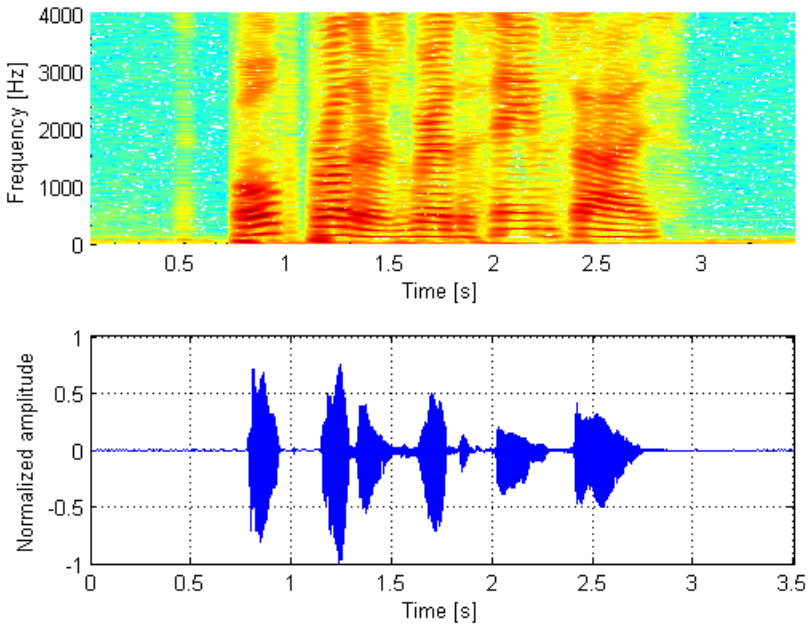

Figure 17: A spectrogram and a waveform plot from of test file "Male 2".

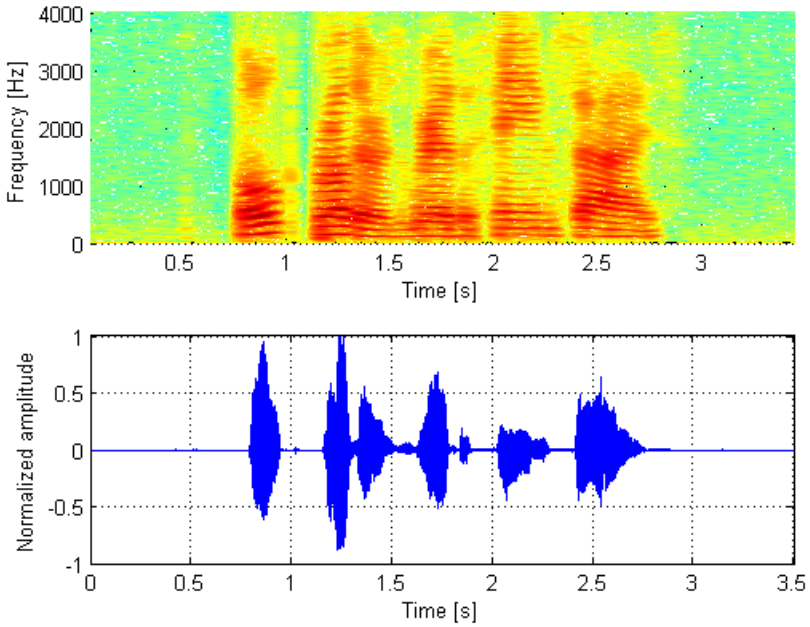

Figure 18: A spectrogram and a waveform plot of test file "Male 2" after encoding to AMR-NB at 4.75 kbps and decoding back to a raw wave.

This confirms that the codec implementations are valid. Test results are given in Table 9.

Table 9: MOS-LQO scores for the tested audio codecs with reference scores from [49].

	AMR NB					G.711	
	12.2kbps	10.2kbps	7.4kbps	5.9kbps	4.75kbps	A-law	μ-law
Female 1	4.26	4.13	4.11	3.83	3.45	4.40	4.49
Female 2	4.24	4.24	4.04	3.84	3.42	4.34	4.37
Male 1	4.37	4.34	4.24	3.99	3.82	4.45	4.51
Male 2	4.34	4.33	4.20	3.97	3.83	4.44	4.50
Average	**4.30**	**4.26**	**4.15**	**3.91**	**3.63**	**4.41**	**4.47**
ITU Ref.	4.20	4.13	4.03	3.83	3.51	4.34	4.47

4.7 Quality and usability of development tools

Tilera offers a complete development environment for developing and debugging applications for its processors. It is named as Tilera Multicore Development Environment (MDE). Tilera MDE is a complete Linux distribution consisting of a kernel, tool chain for building applications, and various libraries. In addition, MDE includes a graphical integrated development environment (IDE). The IDE is based on Eclipse, which is a commonly used platform for IDEs and is widely known among software developers.

4.7.1 Build tools

MDE includes C- and C++ -compilers. The compilers are based on version 4.4.3 of the GCC-compiler and are compliant with ANSI-C and ANSI-C++.

There are several ways to build, run and debug applications within the MDE. First one is to use a launch configuration within the IDE. It offers a graphical dialog-based setup for configuring runs/debug sessions for applications.

Another way of running applications is to invoke commands directly from the host computer's shell application. This is done by using an application called tile-monitor, which is the interface in sofware level to communicate between the TILEPro64-evaluation board and the host computer. Tile-monitor is actually run always when the evaluation board is operated from a host PC. In other words, all other methods of running applications rely on calling the tile-monitor. The behaviour of

the MDE during the runs is determined by the arguments given to tile-monitor. For example, debugging can be enabled via tile-monitor arguments.

Standard Linux makefiles can also be used. This means that different rules for building applications can be determined in the makefile and the build process can then be invoked by selecting a suitable make option. As the build tools of MDE are based on standard Linux tools, they were found to be functional and intuitive to use.

4.7.2 Parallel programming tools

Tilera Medialib comes with a runtime C-library called Tilera task run-time framework (TTR). TTR is used to handle the distribution of computational task between the threads of execution. One or more threads can run on a single tile. A tile refers to an individual processing core in Tilera processors. From the programmer's point of view, TTR hides the actual scheduling and execution of individual tasks. Tasks are the units of execution that are run in parallel. [55]

TTR allocates resources from the processor and offers the available resources to the application as a worker pool. This worker pool is used at runtime to execute single units of execution which in this context, are called tasks. After the worker pool has been initialized, a developer has to assign certain function to a task. This is done via a single function call. When a function has been assigned to a task, the task can be started by another function call. Now the function will be run using all the resources available in the worker pool. [55]

To summarize from the software developer's point of view, the distribution of computational tasks is done by assigning TTR tasks to functions that are desired to be run in parallel. Load balancing is done automatically by the TTR. This type of approach is simple and can easily be utilized by a developer who is not an expert in parallel programming. [55]

4.7.3 Debugging

Tilera MDE includes a version of the gdb-debugger which can be assigned to an individual process. There can be multiple debuggers if there are multiple processes. The debuggers follows the processes that can be executed in several threads in

parallel. This means that the debugger is always attached to a process regardless of which thread it is executed in.

In the graphical IDE, the distribution of threads within a given process is shown in a grid that represent the individual tiles. An example of the grid view is shown in Figure 19. When a breakpoint is hit, the grid view changes color and indicates the thread that was running while the breakpoint was hit. In addition, the tile that the thread was running on can be seen from the grid view. An example of the grid view when a breakpoint has been hit is shown in Figure 20. The debugging tools were found to be functional and the grid view provided an intuitive way of visualizing the distribution of the threads.

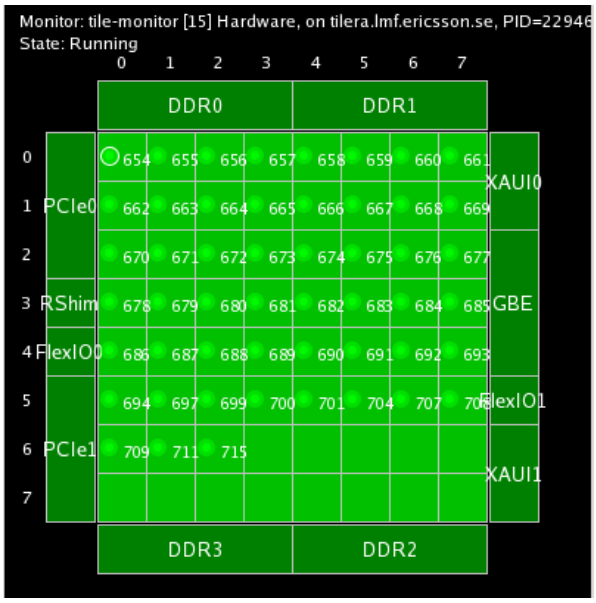

Figure 19: Example of a grid view in MDE while an application is running in 51 threads.

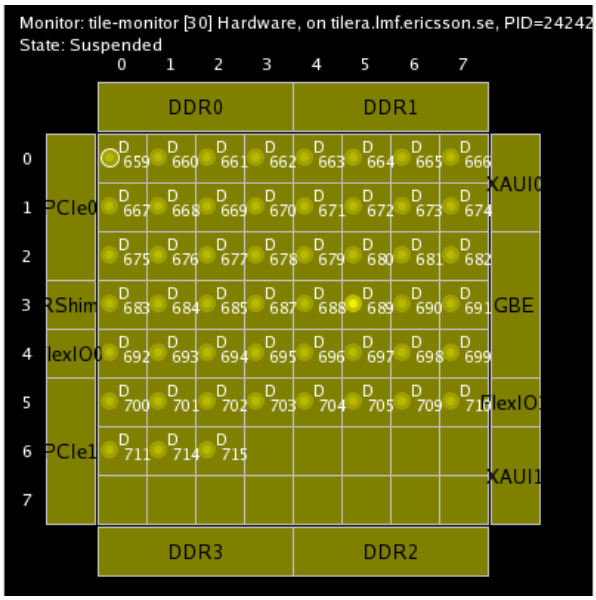

Figure 20: Example of a grid view in MDE when a thread (689) has hit a breakpoint and therefore has been suspended. The thread that caused the suspend-state is marked with a slightly lighter dot.

5 Tilera, Texas Instruments, Freescale – a comparison of multicore solutions

The purpose of this chapter is to compare Tilera multicore processors with current multicore offerings from two major companies (Texas Instruments, Freescale Semiconductor). The current multicore platforms from each vendor are briefly reviewed based on publicly available material and the observed differences in architectures and tools are discussed in a high level.

5.1 Tilera

Tilera Corporation is a semiconductor company headquartered in San Jose, California in the United States. It was founded in 2004 and it is concentrated on developing scalable multicore platforms. [52]

TILE-Gx, TILEPro and TILE64

Currently there are three models available in the TILE-Gx series, including TILE-Gx3036, TILE-Gx8016 and TILE-Gx8036. The last two digits in the model number indicate the numbers of tiles available in the device. The Gx-series represents Tilera's most recent processor techology. Tilera states that the single tile performance of the Gx-series is approximately 2.5 times higher than its predecessor, the TILEPro-series [53]. TILEPro -processors are homogeneous multicore devices that include either 36 or 64 tiles. TILE64 is also a homogeneous multicore device, including 64 tiles. TILE64 was the first model introduced by Tilera in 2007 [52].

An individual tile consists of a general purpose processor core, a cache memory and a switch element that connects the tiles together via an internal network. TILE-Gx8016 and TILE-Gx8036 include coprocessors for network security algorithms and data compression/decompression algorithms. This set of coprocessors is named as the MiCA^{TM}engine. Another coprocessor found in the TILE Gx8000 series is named as mPIPE. This coprocessor is designed for network packet processing and classification. A block diagram of the TILE-Gx8036 platform is shown as an example in Figure 21. Selected features of the Tilera processors are given in Table 10. [56]

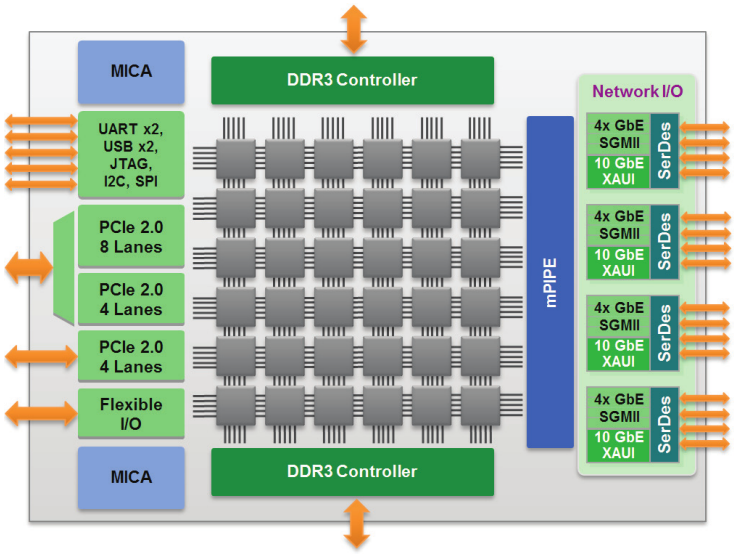

Figure 21: Block diagram of the Tilera TILE-Gx8036. Adopted from [56], courtesy Tilera Corporation.

Table 10: Summary of selected features of the Tilera processors.

Model	GPP	Coprocessors
TILE-Gx8036	36xTILE-Gx	2xMiCA,mPIPE
TILE-Gx8016	16xTILE-Gx	1xMiCA,mPIPE
TILE-Gx3036	36xTILE-Gx	-
TILEPro64	64xTILEPro	-
TILEPro32	36xTILEPro	-
TILE64	64xTILE64	-

Multicore architecture

Individual processor cores in Tilera's Tile^{TM}architecture are arranged in a rectangular array. Tilera's multicore architecture is based on a communication engine named as iMesh^{TM}, that connects the tiles together. The iMesh itself is an array of switch engines and networks that connects them and the I/O devices together. Each tile contains a switch engine and each switch engine is physically connected to neigboring tiles. The tiles that are adjacent to I/O devices connect directly to them.

There are several networks in the iMesh. One of the networks is visible to the programmer and others are used in the background to improve efficiency and speed of data transfers between the tiles, I/O devices and memory. The iMesh is designed to be scalable in a way that the size of the grid of tiles can be increased in future

models without changing the architecture. The structure of the networks in the iMesh illustrated in Figure 22. [57]

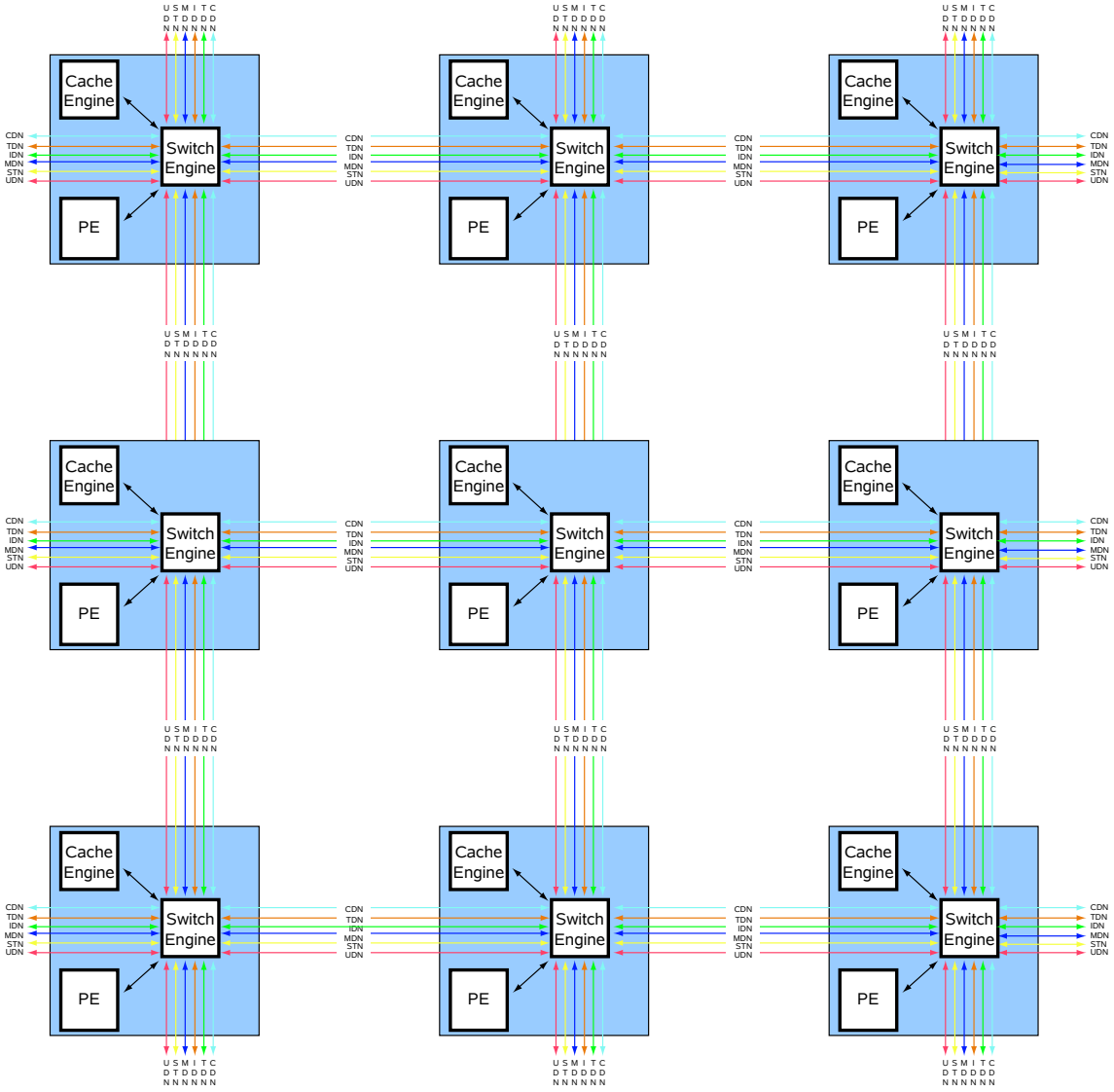

Figure 22: The structure of the networks in the iMesh. Different networks in the Figure are: User Dynamic Network (UDN), I/O Dynamic Network (IDN), Memory Dynamic Network (MDN), Coherence Dynamic Network (CDN) and Tile Dynamic Network (TDN). Adopted from [58], courtesy Tilera Corporation.

5.2 Texas Instruments

Texas Instruments (TI) is a major provider of semiconductor devices. It is headquartered in Dallas, Texas in the United States. Currently TI employs approximately 35300 people worldwide and it develops and manufactures products in areas of analog electronics, embedded systems and wireless technology. [59]

Da Vinci™Digital Media Processors

Texas Instruments Da Vinci™processors are heterogeneous multicore chips that include ARM general purpose processor and DSPs, along with coprocessors for hardware support in multimedia coding. At the time of writing this thesis, there are nine series of Da Vinci processors. Each series contains several models with varying properties. All models in DM816x, DM814x, DM646x series, and two models in DM3x series (DM368, DM365) include one or more High Definition Video Coprocessors (HDVICP). These are designed to handle video coding routines, for example H.264 encoding and decoding. For example, in DM8168 there are three HDVICPs. Texas Instruments states that DM8168 can encode or decode up to three simultaneous streams of H.264 video at 1080p, running at 60 frames per secons (fps). In other words, as the video coding in Da Vinci™processors is done in HDVICPs, one HDVICP can encode or decode one 1080p video stream at 60 fps. Selected features of Da Vinci™processors are listed in Table 11. [60]

Table 11: Selected features of Texas Instruments Da Vinci -platforms. MIPS and MMACS figures are given for the fastest model of the series [60].

Family	GPP	GPP MIPS	DSP	DSP MMACS	Coprocessors
DM816x	ARM Cortex-A8	2400	1 C674x	12000	2-4
DM814x	ARM Cortex-A8	2400	0-1 C674x	6000	1-2
DM64x	-	-	1 C64x	5760	-
DM646x	ARM9	500	1 C64x	8000	6
DM644x	ARM9	405	1 C64x	6480	2-5
DM643x	-	-	1 C64x	5600	2-4
DM3x	ARM9	432	-	-	0-2
DM37x	ARM Cortex-A8	2000	1 C64x	6400	-
OMAP35x	ARM Cortex-A8	1440	1 C64x	4160	-

C6000™Multicore DSP

C6000 series includes both homogeneous and heterogeneous multicore chips. DSPs embedded in this series of chips are either fixed point (C64x series) or hybrid (C66x), that offer both fixed and floating point instructions.

C6670 is a heterogeneous chip that includes four C66x DSPs and sixteen coprocessors for hardware support in networking functions. These coprocessors are:

- 4 Viterbi Decoder 2 Coprocessors (VCP2). These are designed to perform 3G standard channel decoding for convolutional codecs such as AMR.
- 4 Turbo Encoder/Decoder Coprocessors (TCP3e, TCP3d). These are programmable coprocessors designed for turbo coding processes needed in for example WDCMA, LTE and HSUPA connectivity. Three of the coprocessors are for decoding and one for encoding.
- 3 Fast Fourier Transform Coprocessors (FFTC). These coprocessors can be used to perform FFT and inverse FFT -operations.
- 3 Transmit/Recieve Accelerator Coprosessors (TAC, RAC). Designed to handle UMTS operations and assist in transferring data from an antenna to the core components. Two coprocessors are available for recieving and one for transmitting.
- 1 Bit Rate Coprocessor (BRC). This can be used for baseband bit processing, meaning the bit operations involved in connectivity standards such as WCDMA and LTE.
- 1 Network coprocessor (NETCP). This consists of functions that are used to increase ethernet performance of the chip.

The other C667x models are heterogeneous chips, containing 1-8 C66x series DSPs and a network coprocessor. C665x models are also heterogeneous. They contain one or two C66x DSPs and three coprocessors (2 VCP2, 1 TCP3d). C6474-x models are heterogeneous, containing three C64x DSPs and two coprocessors (VCP2, TCP2). C6472-x models are homogeneous, containing six C64x DSPs.

A block diagram of the C6670 is shown in Figure 23 as an example of the C6000 series architecture. Other than the number of DSPs and the number and type of coprocessors, the architecture of other C66x series is similiar. Summary of features of the C6000 series devices is given in Table 12.

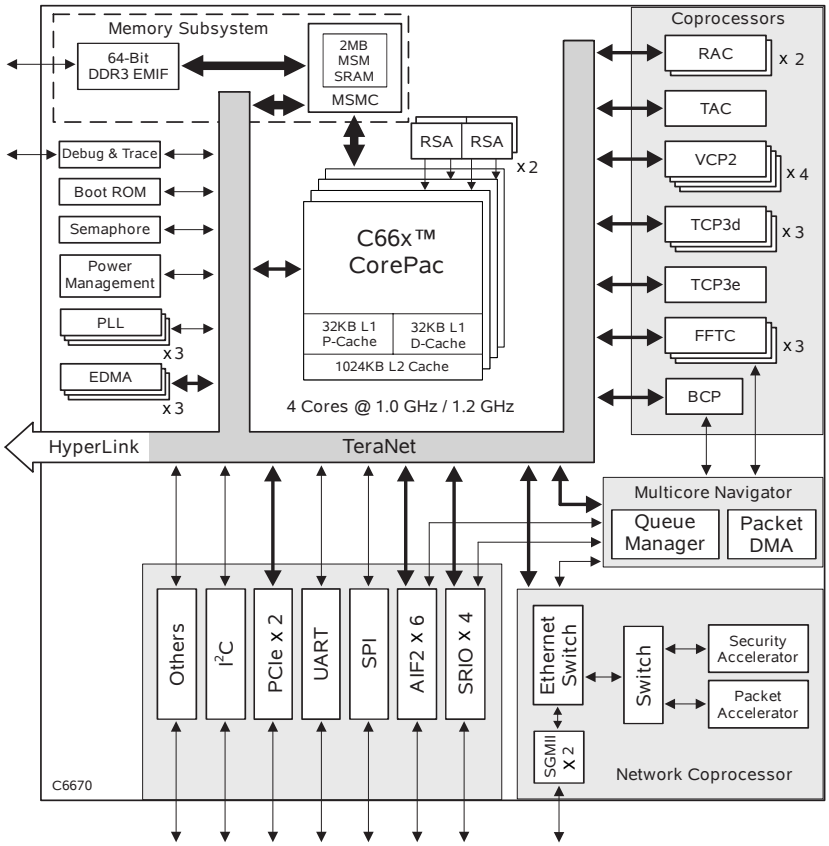

Figure 23: A block diagram of the C6670 architecture. Adopted from [61], courtesy Texas Instruments.

Table 12: Selected features of Texas Instruments C6000-series devices.

Model	DSP	DSP MMACS	Coprocessors
C6678	8xC66x	320000	1
C6674	4xC66x	160000	1
C6672	2xC66x	80000	1
C6671	1xC66x	40000	1
C6670	4xC66x	153000	16
C6657	2xC66x	80000	3
C6655	1xC66x	40000	3
C6474-x	3xC64x	28800	2
C6472-x	6xC64x	33600	-

Multicore architecture

TI's multicore platforms use a packet-based infrastructure that handles data movement and controls the interoperation of the different processor cores. This infrastructure is named as KeyStone architecture and the current version of it is KeyStone

II. Two of the main elements in the KeyStone architecture are named as Multicore Navigator and TeraNet.

Multicore Navigator is a packet-based messaging system that handles the distribution of computational tasks to appriprate hardware resources. The individual DSP cores, coprocessors, I/O devices and memory elements are connected via a switch fabric that forms the TeraNet. [62]

Development tools

TI offers an SDK called Code Composer Studio™(CCStudio), which has extensions to provide control over the various coprocessors in heterogeneous multicore platforms. The SDK is based on Eclipse and the debugging tools are based on the Gnu Project Debugger (GDB). TI also offers a complete Linux distribution for developing applications for its multicore platforms. It includes the CCStudio along with third party plug-ins for a host computer, and Linux kernel, drivers and sample applications for the multicore devices. [63]

TI's compilers for C66x series support the OpenMP API. OpenMP API is an application program interface that has been developed jointly by a group of computer hardware and software vendors, including TI [64]. OpenMP handles the distribution of computational tasks into parallel execution. It can be used to migrate existing sequential C/C++ code into parallel by using compiler directives (pragmas). These directives can be added incrementally into code parts that would benefit from parallel execution [65]. [63]

5.3 Freescale

Freescale Semiconductor Inc. is a global manufacturer of semiconductor devices. It is headquartered in Austin, Texas in the United States and it employs 18000 people in 21 countries. Target markets for Freescale are automotive, consumer, industrial and networking. [66]

QorIQ Qonverge

The QorIQ Qonverge series of processors are heterogeneous multicore platforms that include StarCore DSPs and Power Architecture® general purpose processors. QorIQ

Qonverge is a part of Freescale's QorIQ-family of processing platforms. The devices in the Qonverge series contain general purpose processor cores and DSP cores in one chip. QorIQ-family contains other series as well, such as QorIQ AMP and QorIQ P. The devices in this series do not contain DSPs and are heavily focused on baseband processing using Power Architecture cores. [67]

Power Architecture technology has been developed as a collaboration between Motorola, IBM and Apple. Power Architecture cores used in the QorIQ platforms are e500 and e6500. The e500 is a 32-bit core that was introduced in early 2000's. The e6500 is a more recently introduced 64-bit dual-threaded core. From a software point of view, a dual-threaded core looks like two cores, although the two virtual cores share same resources. The e6500s are manufactured using 28 nm technology, whereas the e500s are manufactured using 45 nm technology. [67]

All devices in the QorIQ Qonverge -series include a version of "Multi-Accelerator Platform Engine for Baseband" (MAPLE-B2P or MAPLE-B3). These are coprocessors designed to handle operations that are required in 3G and 4G connectivity standards such as WCDMA and LTE. Processing elements in MAPLE-B2P are [67]:

- Fourier transform processing element (eFTPE).
- Turbo/Viterbi decoding processing element (eTVPE).
- Downlink encoding processing element (DEPE). This can be used for data rate matching between different parts of the platform and also for turbo encoding algorithms.
- Chip rate processing element (CRPE). This can be used for spreading/despreading and scrambling/descrambling operations required in the WCDMA standard.
- Equalization processing element (EQPE). This performs Multiple-Input-Multiple-Output (MIMO) equalization operations required when using multiple antennaes.
- Physical downlink and uplink processing elements (PDPE, PUPE). These can be used to perform routine operations related to interfacing with antennaes in 3G/4G connectivity.

MAPLE-B3 includes some additional processing elements and mostly new versions of the processing elements listed above. The additional coprocessors in MAPLE-B3 are [67]:

– CRC check and insertion processing element (CRCPE).

– Turbo encoding processing element (TCPE).

Freescale QorlQ platforms are divided into three series, including the G series, the B series and the BSC series. As an example, a block diagram of the B4860 is given in Figure 24.

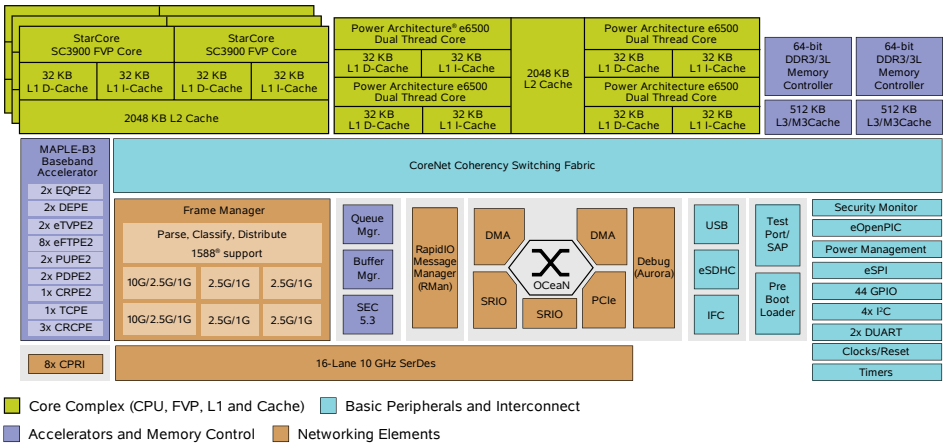

Figure 24: A block diagram of the Freescale QorIQ B4860 architecture. Adopted from [68], courtesy Freescale Semiconductor Inc.

SC3900 DSPs are vector processors, meaning that computational operations can be executed on arrays of numbers instead of individual numbers. This kind of array operations are common in multimedia processing. The BSC series includes two models, BSC9131 and BSC932. The BSC9131 has one e500 general purpose processor and one SC3850 DSP, while the BSC932 includes two of each. A summary of features of the Freescale QorlQ Qonverge -series devices is given in Table 13. [67]

Table 13: Summary of selected features of the Freescale QorIQ Qonverge -series platforms. [67] [68]

Model	GPP	GPP MIPS	DSP	DSP MMACS	Coprocessors
B4860	4xe6500	43200	6xSC3900	230000	MAPLE-B3
B4420	2xe6500	21600	2xSC3900	77000	MAPLE-B3
BSC9132	2xe500	4630	2xSC3850	16000	MAPLE-B2P
BSC9131	1xe500	2315	1xSC3850	8000	MAPLE-B2P
G1110	1xe500	2315	1xSC3850	8000	-

StarCore MSC8x Multi-core DSPs

Freescale's current homogeneous multicore platforms are named as MSC8x, although the MSC8x-series also includes heterogeneous platforms that utilize coprocessors for hardware accelerated networking. The network coprocessors used in this series are named as MAPLE-B and MAPLE-B2 and they include the following processing elements [69]:

- Fourier Transform processing element (FTPE).
- Turbo/Viterbi encoding processing element (TVPE).
- CRC check and insertion processing element (CRCPE).
- Chip rate processing element (CRPE). MAPLE B2 only.

Some models in the MSC8x-series include a Security Engine Coprocessor (SEC) that is designed to process security algorithms related to for example SSL/TLS and LTE. Additionally, some models include a QUICC Engine coprocessor which is designed to handle interoperation between different communications protocols [70]. DSP cores used in the series are SC140, SC3400, SC3850. A larger number in the name indicates a newer model of the DSP core. A summary of features of the Freescale MSC8x -series devices is given in Table 14. [69]

Table 14: Summary of selected features of model families in the Freescale StarCore -series of multicore platforms [67] [68].

Family	DSP	DSP MMACS	Coprocessors
MSC825x	2-6xSC3850	48000	1
MSC815x	2-6xSC3850	48000	2-3
MSC814x	4xSC3400	16000	1
MSC812x	4xSC140	8000	2

Multicore architecture

Freescale's multicore architecture in QorIQ-family of devices is built on CoreNet^{TM}switch fabric, which makes the necessary connections between different processing cores and I/O devices. StarCore DPSs use a different switch fabric. This is named as CLASS, and it is less complex than the CoreNet. Therefore, the CLASS switch fabric takes also less silicon space than the CoreNet. [71]

Development tools

Freescale offers an Eclipse-based IDE named as "CodeWarrior" for developing and debugging applications for StarCore DSPs. Additionally, a royalty-free licence is given for Freescale customers for a real-time operating system called SmartDSP OS. The SmartDSP OS is provided with the IDE as a reference code. It provides the tools to handle division of computational tasks for parallel execution. [72]

Optimized multimedia codecs for StarCore and QorIQ platforms are offered by a third party company called Tata Elxsi, meaning that Freescale does not provide optimized multimedia codec implementations themselves. Tata Elxsi offers a variety of optimized multimedia codecs and a generic media framework for Freescale DSP platforms. [73]

5.4 Summary

Current multicore platforms from Tilera, TI and Freescale and have been reviewed, based high-level descriptions found in publicly available material. Raw MIPS and MMACS figures given in previous sections are intended to offer only informative statistics that can be used to compare the performance of the individual product families of a given vendor. "Apples-to-apples" comparison of performance of the reviewed platforms is not possible based on these figures because of the differences in istructions sets and architectures. A unified benchmark would be required for such comparison.

Freescale offers a wide line of multicore platforms. At the moment, the focus seems to be in baseband processing and therefore the coprocessors included are heavily specialized on algorithms used in 3G/4G baseband processing. The most recent DSP core, the SC3900 offers a competitive performance but at the time of writing, dedicated DSP chips utilizing the SC3900 core were not available. The SC3900 cores are only present in QorIQ -series devices. From an application point of view, Freescale platforms include a lot of coprocessors that are not directly meant for multimedia processing. On the other hand, some processing elements such as the Fourier transform processing element could be utilized in multimedia coding algorithms that require DFT/IDFT operations. Coprocessors that are not needed can be turned off to reduce power consumption.

TI offers both heterogeneous and homogeneous multicore platforms. TI Da Vinci

platforms include coprocessors that are designed for digital media processing such as H.264 coding. TI's C66x series of multicore DSPs offer both fixed and floating point instructions sets in one device, while Freescale DSPs include fixed-point instructions sets. They offer the highest computational performance in TI's DSPs when raw MMACS figures are concerned.

An Eclipse-based IDE and multicore debugging tools are offered for all reviewed platforms. From a developer's point of view, the support for OpenMP API increases the options for handling the parallel programming of TI's C66x series DSPs.

TI is currently the only vendor of the reviewed companies offering dedicated coprocessors for video coding. All vendors offer optimized software implementations of common multimedia codecs for their platforms, although Freescale has outsourced codec development activities to Tata Elxsi.

Multicore architecture of the Tilera platforms is different compared to the architecture solutions of the two major vendors. While all use a switch fabric to connect the different processing cores, memories and I/O devices together, only Tilera processors include a switch element in each core. This eliminates the dependency of a centralized bus architecture and improves the scalability of the architecture.

6 Conclusions and further work

In the future, the traditionally separate fields of speech and music coding seem to be converging. From the point of view of telecommunications, this follows the general trend of convergence in networks and network services. Opus and USAC are examples of these new types of codecs that combine features from the traditional speech and music codecs, although USAC might introduce too much latency for real time applications.

Sound quality is an important factor when delivering multimedia services. It is important that in the future, the audio codecs used in telecommunications are capable of delivering sound at sufficient quality, reagardless of whether it is speech, music or any other kind of audio.

Tilera TILEPro64 -multicore platform and a related software library Medialib alpha2 has been reviewed in terms of performance in multimedia coding. In addition, the usability and quality of the development tools have been assessed informally. In order to verify the performance, test applications were developed to run encoding and decoding using two video codecs (H.264, H.263+) and three audio codecs (G.711 μ-law and A-law, AMR-NB).

Test results showed that the performance of the tested multicore platform and media codec library approximately matches the performance figures given by the vendor of the multicore platform. For a single stream of video data, FPS results grew approximately in a linear fashion up to a certain point that depends on the codec and resolution. Beyond that saturation point, increasing the number of cores does not increase the performance significantly. The behaviour was found to be similiar for encoding and decoding. As expected, H.264 with the limited implementation of high profile was found to be computationally the most demanding of the tested multimedia codec implementations. The video codec implementations were found to be valid by informally inspecting the encoded video files with a media player software.

Audio codecs included in the test were significantly less complex than the video codecs and it was found that using multiple cores does not increase the coding performance for a single simultaneous stream. The audio codec implementations were found to be valid by running PESQ-analysis for the encoded/decoded files and by comparing MOS-LQO scores with reference results provided by ITU.

The tested software library was not a release version and it included limitations in the high profile option of H.264. In addition, further developed versions of the Tilera multicore platforms have been released. When the Medialib reaches a release version, it would be informative to test its performance again in order to gain final results for performance in H.264 high profile coding.

The current multicore platform offerings of two major vendors (TI, Freescale) were reviewed based on publicly available material and compared to Tilera's current multicore platform offerings. Development tool offerings of each vendor were found to be similiar, including an Eclipse-based IDE with debugging tools capable of handling multiple cores and the associated parallel execution of tasks. In terms of communication between individual processing cores, Tilera's hardware architecture was found to have some differences when compared to the two major vendors.

As a further study, TI's and Freescale's multicore products could be tested using the same test applications and test data that were used while testing Tilera's multicore product. This would make it possible to determine their relative performance.

References

[1] J. Tebo, "The early history of telecommunications: The first hundred years," *Communications Society*, vol. 14, no. 4, pp. 12–21, 1976.

[2] K. Granlund, *Tietoliikenne*. Docendo Finland Oy, Apr. 2003.

[3] Ficora, "Broadband and telephone services, Statistical review July-December 2011." Ficora Market Review, February 2012.

[4] OECD, "Machine-to-Machine Communications: Connecting Billions of Devices." OECD Digital Economy Papers, No. 192, OECD Publishing. `http://dx.doi.org/10.1787/5k9gsh2gp043-en`, 30 Jan. 2012, Accessed 12 June 2012.

[5] 3GPP, "3GPP website." `http://www.3gpp.org/`, Accessed 11 June 2012.

[6] ETSI, "3GPP TS 123 228 V10.7.0." Technical Specification, January 2012.

[7] D. Khedher, R. Glitho, and R. Dssouli, "Media handling for multimedia conferencing in multihop cellular networks," *Network, IEEE*, vol. 23, pp. 35–42, March 2009.

[8] Ericsson, "IMS Control and Media." Company website, `http://www.ericsson.com/ourportfolio/products/ims-control-and-media`, Accessed: 9 September 2012.

[9] M. M. Hannuksela, *Error-Resilient Communication Using the H.264/AVC Video Coding Standard*. PhD thesis, Tampere University of Technology, March 2009.

[10] H. S. Black and J. O. Edson, "Pulse code modulation," *Transactions of the American Institute of Electrical Engineers*, vol. 66, pp. 895–899, January 1947.

[11] R. M. Gray, "A Survey of Linear Predictive Coding: Part I of Linear Predictive Coding and the Internet Protocol," *Foundations and Trends in Signal Processing*, vol. 3, no. 3, pp. 153–202, 2009.

[12] A. Cinnéide, "Linear Prediction -The Technique, Its Solution and Application to Speech." Dublin Institute of Technology, Online PDF, `http://eleceng.dit.ie/papers/92.pdf`, August 2008, Accessed 26 July 2012.

[13] J. Bradbury, "Linear Predictive Coding." Online PDF, `http://my.fit.edu/~vKepuska/ece5525/lpc_paper.pdf`, December 2000, Accessed 26 July 2012.

[14] T. Painter and A. Spanias, "Perceptual coding of digital audio," *Proceedings of the IEEE*, vol. 88, pp. 451–515, April 2000.

[15] Y. Wang, and M. Vilermo, "The modified discrete cosine transform: Its implications for audio coding and error concealment," in *Audio Engineering Society Conference: 22nd International Conference: Virtual, Synthetic, and Entertainment Audio*, June 2002.

[16] C. Cellier, P. Chenes and M. Rossi, "Lossless audio bit rate reduction," in *Audio Engineering Society Conference: UK 9th Conference: Managing the Bit Budget (MBB)*, May 1994.

[17] D. Huffman, "A method for the construction of minimum-redundancy codes," *Proceedings of the IRE*, vol. 40, pp. 1098–1101, September 1952.

[18] ITU-T, "Codecs for Videoconferencing Using Primary Digital Group Transmission." Recommendation H.120, March 1993.

[19] C. Lima et al, "High definition video broadcast over core ip networks," in *Optical Fiber Communication Conference, 2006 and the 2006 National Fiber Optic Engineers Conference. OFC 2006*, p. 10, March 2006.

[20] R. Aalmoes, "Video compression techniques over low-bandwidth lines," Master's thesis, Twente University, 27 August 1996.

[21] Berkeley Design Technology, "Introduction to video compression." EETimes online article, `http://www.eetimes.com/design/signal-processing-dsp/4013042/Introduction-to-video-compression`, 13 April 2006, Accessed 9 June 2012.

[22] ITU-T, "Pulse Code Modulation (PCM) of Voice Frequencies." Recommendation G.711, 1972.

[23] ITU-T, "Low-complexity, full-band audio coding for high-quality, conversational applications." Recommendation G.719, June 2008.

[24] ITU-T, "7 kHz Audio-coding within 64 kbit/s." Recommendation G.722, 1988.

[25] ITU-T, "Coding of speech at 8 kbit/s using conjugate-structure algebraic-code-excited-linear prediction (CS-ACELP)." Recommendation G.729, January 2007.

[26] ETSI, "3GPP TS 126 090 V.3.1.0." Technical Specification, January 2000.

[27] ETSI, "3GPP TS 26.190 V5.1.0." Technical Specification, December 2001.

[28] ETSI, "3GPP TS 06.51 V.4.2.0." Technical Specification, December 2000.

[29] ITU-T, "Advanced video coding for generic audiovisual services." Recommendation H.264, May 2003.

[30] ITU-T, "Video coding for low bit rate communication." Recommendation H.263, March 1996.

[31] Texas Instruments, "A-Law and mu-Law Companding Implementations Using the TMS320C54x." Application Note SPRA163A, December 1997.

[32] J. Curcio and I. Kalliokulju, "AMR mode selection enhancement in 3G networks," *Multimedia Tools and Applications*, vol. 28, no. 3, pp. 259–281, 2006.

[33] ETSI, "3GPP TS 126 071 V5.0.0." Techical Specification, June 2002.

[34] ETSI, "3GPP TS 126 290 V10.0.0." Technical Specification, April 2011.

[35] ITU-T, "Video coding for low bit rate communication." Recommendation H.263, January 2005.

[36] ETSI, "3GPP TS 126141 V10.0.0." Techical Specification, April 2011.

[37] S. Warner and J. Bhal, "The role of multicore in the evolution of communications." Texas Instuments White Paper, April 2011.

[38] Franhofer IIS, "Fraunhofer IIS and VoiceAge provide reference for upcoming MPEG audio codec." Press Release, `http://www.iis.fraunhofer.de/en/pr/presse/2008/09/voiceage.jsp`, 9 September 2008, Accessed 17 July 2012

[39] M. Neuendorf et al, "MPEG unified speech and audio coding - the ISO/MPEG standard for high-efficiency audio coding of all content types." Convention Paper, 132nd AES Convention, April 2012.

[40] S. Quackenbush, "MPEG unified speech and audio coding," in *Audio Engineering Society Conference: 43rd International Conference: Audio for Wirelessly Networked Personal Devices*, September 2011.

[41] A. Raake et al, "How to talk about speech and audio quality with speech and audio people," *J. Audio Eng. Soc*, vol. 60, no. 3, pp. 147–155, 2012.

[42] JM. Valin, K. Vos and T. Terriberry, "Definition of the Opus Audio Codec." IETF Internet Draft (work in progress), `http://tools.ietf.org/html/draft-ietf-codec-opus-04`, June 2012, Accessed 18 July 2012.

[43] C. Hoene et al, "Summary of Opus listening test results." IETF Internet Draft (work in progress), `http://tools.ietf.org/html/draft-ietf-codec-results-01`, 1 May 2012, Accessed 18 July 2012.

[44] A. Rämö and H. Toukomaa, "Voice Quality Characterization of IETF Opus Codec." Interspeech 2011, Florence, Italy, Aug. 2011.

[45] A. Roy, J. Xu, and M. Chowdhury, "Multi-core processors: A new way forward and challenges," in *International Conference on Microelectronics 2008*, pp. 454–457, December 2008.

[46] ITU-T, "Perceptual evaluation of speech quality (PESQ): An objective method for end-to-end speech quality assessment of narrow-band telephone networks and speech codecs." Recommendation P.862, November 2005.

[47] T. Siren, "Test tool for measuring objective speech quality in mobile media gateway multiparty call," Master's thesis, Aalto University, School of Electrical Engineering, 1 December 2011.

[48] ITU-T, "Mean Opinion Score (MOS) terminology." Recommendation P.800.1, July 2006.

[49] ITU-T, "Application guide for objective quality measurement based on Recommendations P.862, P.862.1 and P.862.2." Recommendation P.862.3, November 2007.

[50] Tilera Corporation, "TILExpress-64, TILExpressPro-64 Card User's Guide." , March 2011.

[51] ITU-T, "Test signals for use in telephonometry." Recommendation P.501, December 2009.

[52] Tilera Corporation, "About Tilera." Company website, `http://www.tilera.com/about_tilera`, Accessed 29 August 2012.

[53] Tilera Corporation, "Tilera Medialib - FAQ Version 1.7." Medialib documentation, 19 March 2012.

[54] VideoLAN, "VLC Player 2.0.3." Open source software, http://http://www.videolan.org/vlc/, Downloaded 25 August 2012.

[55] Tilera Corporation, "Tilera Task Runtime (TTR) Overview." Medialib documentation, 31 May 2011.

[56] Tilera Corporation, "TILE-Gx8036 Processor Specification Brief." Online PDF, http://www.tilera.com/sites/default/files/productbriefs/Tile-Gx%203036%20SB012-01.pdf, 3 May 2012, Accessed 29 August 2012.

[57] Tilera Corporation, "Tile Processor User Architecture Manual." Online PDF, http://www.tilera.com/scm/docs/UG101-User-Architecture-Reference.pdf, 29 March 2011, Accessed 29 August 2012.

[58] Tilera Corporation, "Tile Processor Architecture Overview for the TILEPro Series." Online PDF, http://www.tilera.com/scm/docs/UG120-Architecture-Overview-TILEPro.pdf, 28 March 2011, Accessed 29 August 2012.

[59] Texas Instruments, "Company." Company website, http://www.ti.com/corp/docs/company/index.shtml, Accessed 28 August 2012.

[60] Texas Instruments, "Embedded Processors" Company website, http://www.ti.com/lsds/ti/dsp/embedded_processor.page, Accessed 2 August 2012.

[61] Texas Instruments, "TMS320C6670 Multicore Fixed and Floating-Point System-on-Chip." Data Manual, Revision D, March 2012.

[62] Texas Instruments, "Comparing TI's TMS320C6671 DSP with ADI's ADSP-TS201S TigerSHARC Processor." White Paper, 6 January 2012.

[63] Texas Instruments, "KeyStone Multicore SoC Tool Suite: One platform for all needs." White Paper, 17 June 2011.

[64] OpenMP ARB, "About the OpenMP ARB and OpenMP.org." Corporate website, http://openmp.org/wp/about-openmp/, 12 August 2012, Accessed 30 August 2012.

[65] Texas Instruments, "OpenMP Programming for TMS320C66x multicore DSPs." Online PDF, http://www.ti.com/lit/pdf/sprt620, 28 August 2012, Accessed 30 August 2012.

[66] Freescale Semiconductor Inc., "About Freescale." Company website, http://www.freescale.com/webapp/sps/site/homepage.jsp?code=COMPANY_INFO_HOME, Accessed 15 August 2012.

[67] Freescale Semicondictor Inc., "QorIQ Processing Platforms." Company website, http://www.freescale.com/webapp/sps/site/homepage.jsp?code=QORIQ_HOME, Accessed 14 August 2012.

[68] Freescale Semiconductor Inc., "Next-Generation Wireless Network Bandwidth and Capacity Enabled by Heterogeneous and Distributed Networks." White Paper, 8 August 2012.

[69] Freescale Semiconductor Inc., "StarCore Digital Signal Processors." Company website, http://www.freescale.com/webapp/sps/site/homepage.jsp?code=STARCORE_HOME, Accessed 15 August 2012.

[70] Freescale Semiconductor Inc., "Introducing Freescale's QUICC Engine Technology." Fact Sheet, 15 June 2007.

[71] Freescale Semiconductor, "Embedded Multicore, An Introduction." Online PDF (requires registration), https://www.freescale.com/webapp/sps/download/preDownload.jsp?render=true, July 2009, Accessed 27 August 2012.

[72] Freescale Semiconductor Inc., "CodeWarrior Development Studio for StarCore v10.1." Fact Sheet, 12 July 2010.

[73] Tata Elxsi, "Tata Elxsi Freescale Partnership." Company website, http://www.tataelxsi.com/htmls/pds/pds_multimedia-freescale.html, Accessed 15 August 2012.